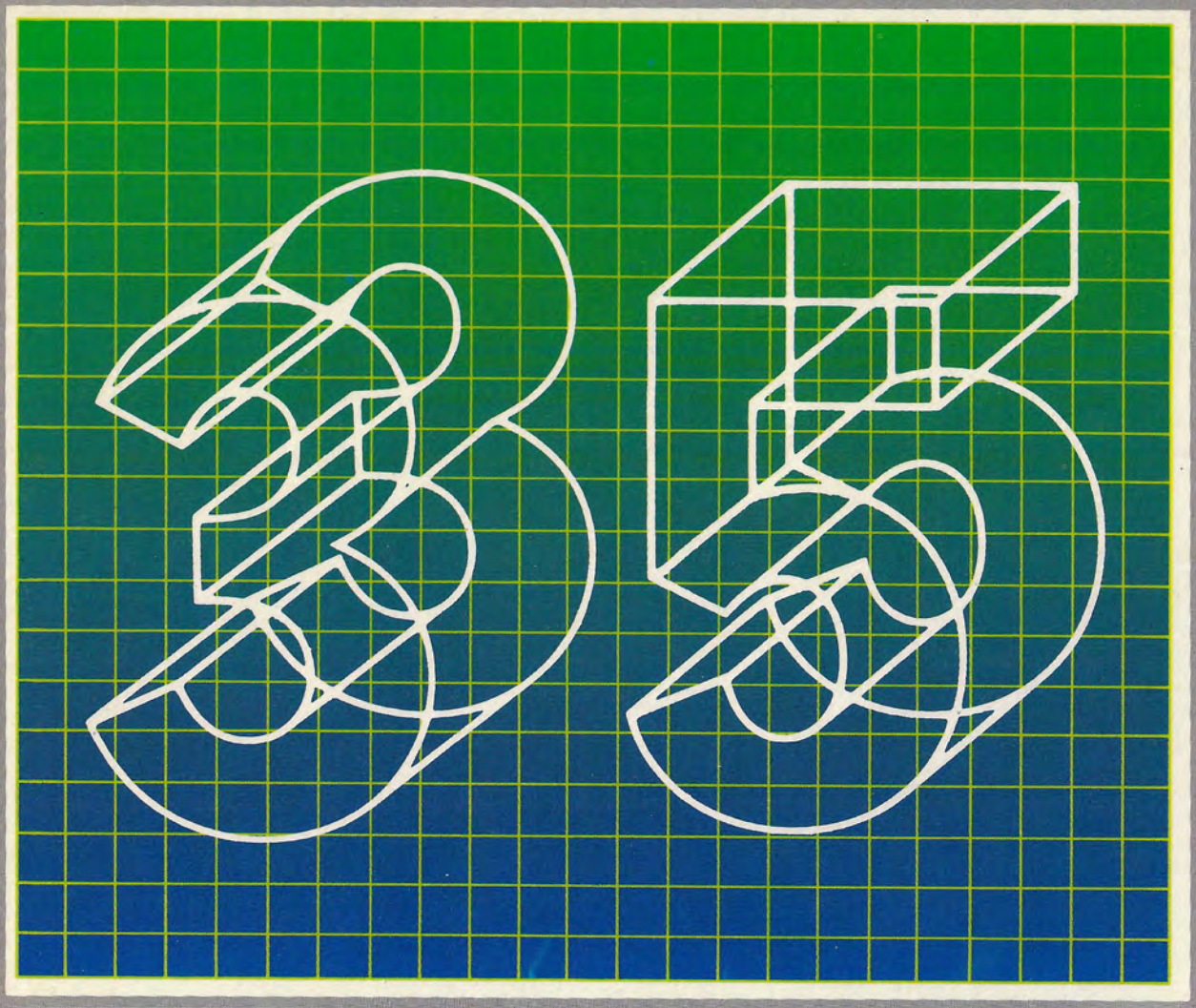
BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



PROGRAMAS PRACTICOS

TRUCOS, BRICOLAGE Y GRAFICOS

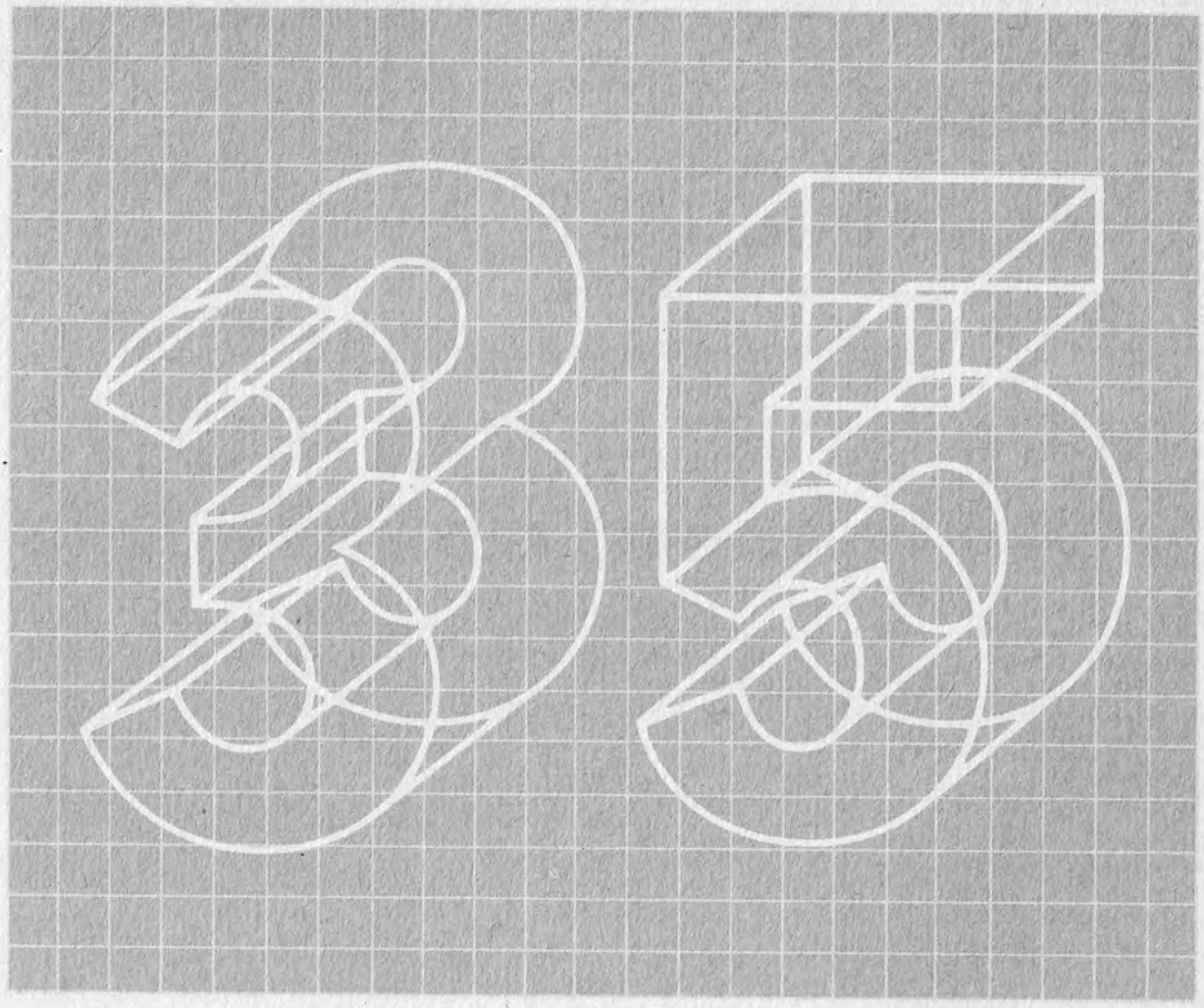
EDICIONES SIGLO CULTURAL

BIBLIOTECA PRACTICA

TALLER DE INFORMATICA

TRUCOS - SPRITES
BRICOLAGE DEL HARD
AULA ABIERTA
LOGO - TERMINOLOGIA

PROGRAMAS VALIDOS PARA AMSTRAD, IBM, SPECTRUM, COMMODORE Y MSX



EDICIONES SIGLO CULTURAL

Una publicación de

EDICIONES SIGLO CULTURAL, S. A.

Director-editor:
RICARDO ESPAÑOL CRESPO.

Gerente:
ANTONIO G. CUERPO.

Directora de producción:
MARIA LUISA SUAREZ PEREZ.

Director de la colección:
JOSE ARTECHE, Ingeniero de Telecomunicación

Diseño:
BRAVO-LOFISH.

Maquetación:
D. S. M.

Dibujos:
JOSE OCHOA

Tomo 35: «Experiencias y prácticas en logo», Carlos Albert, Técnico de Infomática; Adoración Llena Jubelo, Técnica de Informática. «Manejo de sprites y elementos gráficos, «Trucos y rutinas básicas», Francisco Morales. Técnico de Informática. «Aprende con el ordenador», AULA DE INFORMATICA APLICADA: Antonio Monje, Licenciado en Veterinaria; Alejandro Marcos, Licenciado en Química. «El Taller del Hardware», Carlos Rey, Ingeniero Industrial. «Pequeña historia de la Informática», «Temas monográficos de vanguardia», «Terminología», Blanca Arbizu, Técnico de Informática.

Ediciones Siglo Cultural, S. A.

Dirección, redacción y administración:
Pedro Teixeira, 8-2.°. Teléf. 455 09 99. 28020 Madrid.

Publicidad:
Gofar Publicidad, S. A. Benito de Castro, 12 bis. 28028 Madrid.

Distribución en España:
COEDIS, S. A. Valencia, 245. Teléf. 215 70 97. 08007 Barcelona.
Delegación en Madrid: Serrano, 165. Teléf. 411 11 48.

Distribución en Ecuador: Muñoz Hnos.

Distribución en Perú: DISELPESA.

Distribución en Chile: Alfa Ltda.

Importador exclusivo Cono Sur:
CADE, S.R.L. Pasaje Sud América, 1532. Teléf.: 21 24 64.
Buenos Aires - 1.290. Argentina.

Todos los derechos reservados. Este libro no puede ser, en parte o totalmente, reproducido, memorizado en sistemas de archivo, o transmitido en cualquier forma o medio, electrónico, mecánico, fotocopia o cualquier otro, sin la previa autorización del editor.

ISBN del tomo: 84-7688-174-6
ISBN de la obra: 84-7688-047-2

Fotocomposición:
ARTECOMP, S. A. Albarracín, 50. 28037 Madrid.

Imprime:
GRAFICAS EMA. (Madrid)

© Ediciones Siglo Cultural, S. A., 1986

Depósito legal: M-43.185-1986
Printed in Spain - Impreso en España.

Suscripciones y números atrasados:
Ediciones Siglo Cultural, S. A.
Pedro Teixeira, 8-2.°. Teléf.: 259 73 31. 28020 Madrid.

Noviembre, 1987

P.V.P. Canarias: 355.—Ptas.

INDICE

EXPERIENCIA Y PRACTICAS EN LOGO 5

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS 14

TRUCOS Y RUTINAS BASICAS 24

EL TALLER DEL HARDWARE 32

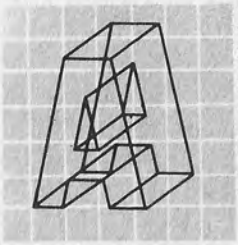
APRENDER CON EL ORDENADOR 39

PEQUEÑA HISTORIA DE LA INFORMATICA 51

TEMAS MONOGRAFICOS DE VANGUARDIA 55

TERMINOLOGIA 59

Parar y continuar



LGUNAS versiones de Logo disponen de dos primitivas con las cuales podemos detener, y posteriormente continuar, la ejecución de un procedimiento.

Se trata de:

— PAUSA

— CONTINUA

Conozcamos el funcionamiento de estas dos nuevas primitivas.

— *PAUSA*

Con esta primitiva podemos detener la ejecución de un procedimiento en el momento que deseemos. Para ello lo único que tenemos que hacer es escribir la orden PAUSA en aquel punto del procedimiento que lo requiera.

La diferencia de poner PAUSA a poner ALTO, es que si utilizamos PAUSA, la ejecución del procedimiento se detendrá hasta que nosotros queramos continuar, y ALTO detiene la ejecución por completo.

La ejecución de un procedimiento se podrá detener bien utilizando la orden PAUSA en el conjunto de las órdenes que lo forman, o bien pulsando la tecla F5 en el momento que lo deseemos.

En ambos casos, recibiremos un mensaje en el que nos avisa que se ha producido una pausa. Este mensaje nos dice también el nombre del procedimiento y la orden en la que se ha producido.

Por ejemplo, si definimos el siguiente procedimiento:

```
? PARA A :V
> ES :V
> ES "
> A :V + 1
> FIN
```

y ahora lo ejecutamos:

```
? A 1
```

en la pantalla empezarán a aparecer todos los números a partir del uno. Si no detenemos la ejecución, el procedimiento no pararía hasta que apareciese un número enormemente grande.

Si en cualquier momento pulsamos F5, en la pantalla aparecerá:

```
PAUSA... EN A
ES :V
A ?
```

Nos dice que se ha producido una pausa en el procedimiento A al ir a ejecutar la orden ES :V.

A continuación aparece el nombre del procedimiento seguido de la interrogación, lo que indica que se está en una pausa y no en el nivel superior.

Si ahora introducimos en el procedimiento la orden PAUSA:

```
? PARA A :V
> ES :V
> ES "
> PAUSA
> A :V + 1
> FIN
```

y lo ejecutamos:

```
? A 1
```

Un procedimiento se puede detener con la primitiva PAUSA.

en la pantalla aparecerá:

```
1
PAUSA... EN A
PAUSA
A ?
```

La pausa se ha producido después que se ha escrito el primer número, en el procedimiento A y al ir a ejecutar la orden PAUSA.

Durante el tiempo que dura la pausa, podemos teclear las órdenes que deseemos de forma interactiva.

La ejecución de un procedimiento no se podrá detener mediante la tecla F10 (ALTO) cuando se esté en una pausa.

La primitiva PAUSA sólo se podrá utilizar dentro de un procedimiento.

— *CONTINUA*

Conozcamos ahora la forma de reemprender la ejecución de un procedimiento que se haya detenido mediante una PAUSA.

Para esto utilizamos la primitiva:

```
? CONTINUA
```

Esta orden podemos también expresarla de forma abreviada:

```
? CO
```

Al introducirla, la ejecución continuará justo en el punto que se haya detenido.

Por ejemplo, si definimos el siguiente procedimiento:

```
? PARA SUMAR :V
> HAZ "A :V + 1
> ES :A
> ES "
> SUMAR :V + 1
> FIN
```

y lo ejecutamos:

```
? SUMAR
```

si una vez pasado un pequeño espacio de tiempo pulsamos F5, en la pantalla aparecerá:

```
2
3
4
5
PAUSA... EN SUMAR
SUMAR ?
```

Si ahora deseamos que continúe la ejecución, introducimos:

```
PAUSA... EN SUMAR
SUMAR ? CO
```

y en la pantalla seguirán apareciendo los números:

```
6
7
.
.
```

y así sucesivamente.

Si en el procedimiento SUMAR introducimos la orden PAUSA:

```
? PARA SUMAR :V
> HAZ 'A :V + 1
> ES :A
> ES '
> PAUSA
> SUMAR :V + 1
> FIN
```

y lo ejecutamos:

```
? SUMAR 1
```

en la pantalla aparecerá:

```
2
PAUSA... EN SUMAR
SUMAR ?
```

si ordenamos que continúe:

```
PAUSA... EN SUMAR
SUMAR ? CO
```

obtendremos:

```
3
PAUSA... EN SUMAR
SUMAR ?
```

y así sucesivamente.

Se realiza una pausa cada vez que se escribe un número.

La primitiva CONTINUA (CO) nunca podrá utilizarse dentro de un procedimiento.

Estas dos primitivas que hemos visto no existen en las versiones de Logo de los MSX ni del SPECTRUM. Sí las podemos encontrar en algunas versiones para los PC-COMPATIBLES.

Otras primitivas

Para presentar textos en la pantalla hemos utilizado, hasta ahora, la primitiva ESCRI-

La primitiva PAUSA sólo puede utilizarse dentro de un procedimiento.

BE; veamos ahora otra primitiva que nos permite también presentar textos en pantalla.

— TECLEA objeto

Presenta en pantalla el objeto especificado sin saltar a la línea siguiente.

Compruébalo con los siguientes ejemplos:

```
? PARA E1
> TECLEA "LIBRO
> ES "S
> TECLEA "CUADERNO
> FIN

? E1
LIBROS
CUADERNO?
```

En la pantalla aparecen juntas las palabras LIBRO y la letra S.

La primitiva TECLEA deja el cursor posicionado al final del objeto que presenta en la pantalla, por eso la S aparece detrás de LIBRO.

Con el último TECLEA del procedimiento E1, podemos observar que el interrogante del LOGO y el cursor quedan situados detrás de la palabra CUADERNO, es decir, no ha saltado de línea.

```
? PARA E2
> ES "EJERCICIOS
> TECLEA "MATEMATICAS
> TECLEA "LENGUA
> ES "FISICA
> ES "GEOMETRIA
> TECLEA "LATIN
> ES "INGLES
> FIN
```

Ejecuta el procedimiento y examina con atención lo que aparece en la pantalla.

```
? E2
EJERCICIOS
METAMATICAS.LENGUA.FISICA
GEOMETRIA
LATIN.INGLES
```

Si el dato que especificamos con la primitiva TECLEA es una lista, ésta se presenta sin los corchetes.

```
? TECLEA [LA CIUDAD]
LA CIUDAD?
```

Si queremos presentar varios objetos con la primitiva TECLEA, debemos poner la primitiva y los objetos entre paréntesis.

(TECLEA objeto1 objeto2...)

Ejemplos:

```
? PARA E3
> ESCRIBE "OBJETOS
> (TECLEA "CASA "JARDIN [SILLA
AZUL] "SILLON)
> FIN

? E3
OBJETOS
CASAJARDINSILLA AZULSILLON?
```

El interrogante del LOGO y el cursor quedan detrás del último objeto de la lista.

```
? PARA E4

> HAZ "A 40
> (TECLEA [LA SUMA ES :] :A) ES "
> FIN

? E4
LA SUMA ES :40
?
```

El interrogante del LOGO queda en la línea siguiente porque hemos incluido la primitiva ES".

Cuadro resumen

— PAUSA (PC-COMPATIBLES)

Detiene, sin llegar a interrumpir, el procedimiento que se está ejecutando en ese momento.

— CONTINUA (CO) (PC-COMPATIBLES)

Permite reemprender la ejecución de un procedimiento previamente detenido con PAUSA.

— TECLEA objeto

(TECLEA objeto1 objeto2...)

Presenta en pantalla el objeto o los objetos especificados sin pasar a la línea siguiente.

Podemos reemprender la ejecución de un procedimiento con la primitiva CONTINUA.

Ejercicios

1. *Este ejercicio servirá para repasar el significado de una serie de palabras inglesas.*

   En un procedimiento se especificarán dos listas:

   a) Una lista con las palabras en inglés que se quieren repasar.

   b) Una lista con las correspondientes traducciones de las palabras de la lista en inglés.

   En la ejecución se mostrará una palabra en inglés y se pedirá su significado en castellano.

   Al finalizar se informará de los aciertos y de los fallos a lo largo de toda la sesión.

2. *Define los procedimientos necesarios para conseguir que las dos listas de palabras, la lista de palabras inglesa y la lista de las respectivas traducciones al castellano, definidas en el ejercicio anterior a través del procedimiento LISTAS, puedan ser introducidas desde el teclado al iniciar cada sesión.*

3. *Define los procedimientos necesarios para realizar el siguiente proceso:*

   — Se solicitará que se introduzca una palabra que no exceda de diez caracteres.
   — Tras pulsar una tecla aparecerá en la pantalla la palabra introducida separados sus elementos dos espacios en blanco.
   — A continuación aparecerán en la parte inferior de cada letra el número de orden que tiene cada una en el abecedario.

4. *Define los procedimientos necesarios para rellenar una zona de la pantalla con un carácter determinado. Para ello se solicitará que se introduzca, la primera y la última columna y la primera y la última fila de la zona. También se solicitará el carácter con el que queremos rellenar.*

5. *¿Son correctas las siguientes órdenes?*

```
— ? PARA A
  > ES AZAR 1000
  > PAUSA
  > CO
  > A
  > FIN
— ? REPITE 10 [PAUSA AV 10 GD 90]
— ? TECLEA "LA "CASA "DE "JUAN
— ? PARA A1
  > HAZ "A 30
  > HAZ "B 40
  > PONCURSOR [4 10]
  > TECLEA [EL RESULTADO DE LA
    SUMA ES :]
  > ES ?A + B
  > FIN
— ? TECLEA ("NO "ESTA [TODO])
```

Solución de los ejercicios

1:

Primero definimos un procedimiento para guardar en la varible A y en la variable B, la lista de palabras en inglés y la lista de palabras en castellano, respectivamente.

Nosotros realizamos este ejercicio con doce palabras.

```
? PARA LISTAS
> HAZ "A [MAN WOMAN HOUSE
  TOOTH FOOT KNIFE LADY BUS
  GLASS BOX DISH YOUNG]
> HAZ "B [HOMBRE MUJER RATON
  DIENTE PIE CUCHILLO SEÑORA
  AUTOBUS VASO CAJA FUENTE JO-
  VEN]
> FIN
```

Con el siguiente procedimiento presentamos la pantalla de trabajo:

```
? PARA P1
> PONCURSOR [9 5]
> ES "ENGLISH
> PONCURSOR [23 5]
> ES "CASTELLANO
> SL
> PONPOS [-130 40]
> BL
> REPITE 2 [AV 40 GD 90 AV 260 GD
  90]
> AV 20 GD 90
> GI 90 AV 20 RE 40
> FIN
```

CONTINUA no se puede utilizar dentro de un procedimiento.

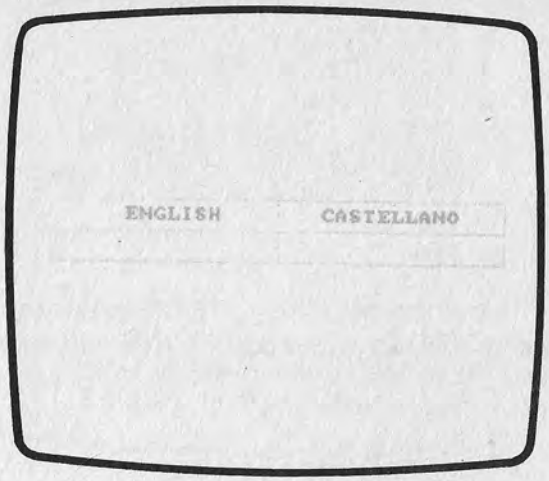


Fig. 1.

En P2 se obtiene, aleatoriamente, la palabra (de la lista de inglés) a preguntar, se muestra en la pantalla y se pregunta su significado en castellano.

```
? PARA P2
> HAZ "I AZAR 13
> SI :I = 0 [HAZ "I 11]
> HAZ "E ELEMENTO :I :A
> HAZ "C ELEMENTO :I :B
> PONCURSOR [8 7]
> ES :E
> PONCURSOR [4 2]
> ES [INTRODUCE LA PALABRA EN
  ESPAÑOL]
> PONCURSOR [23 7]
> HAZ "P LL
> FIN
```

En P3 se comprueba si la traducción es correcta o no.

Si es correcta, se incrementa en 1 la variable que controla los aciertos y se escribe en la pantalla la palabra CORRECTO.

Si no es correcta, se incrementa en 1 la variable que controla los fallos, se escribe en la pantalla la palabra INCORRECTO y la respuesta correcta.

```
? PARA P3
> HAZ "C LISTA :C
> PONCURSOR [10 10]
> SI :P = :C [HAZ "N :N + 1 ES
  "CORRECTO]
  [HAZ "M :M + 1 ES "INCORRECTO
  PONCURSOR [2 12]
  TECLEA [LA PALABRA CORRECTA
  ES :] ES :C]
> FIN
```

Para preguntar si se desea otra palabra, definimos el procedimiento P4.

Si la contestación es (S), se ejecuta el procedimiento TO; si no, se retorna al procedimiento principal.

```
? PARA P4
> PONCURSOR [10 18]
> ES [¿OTRO PALABRA? (S/N)]
> HAZ "T LC
> SI :T = "S [TO]
> SI :T = "N [ALTO]
> FIN
```

El procedimiento TO realiza las llamadas a los procedimientos detallados hasta ahora.

```
? PARA TO
> BP
> BT
> P1
> P2
> P3
> P4
> FIN
```

FINAL se ejecuta cuando ya no se desean traducir más palabras, es con este procedimiento con el que se presenta el número de aciertos y número de fallos.

```
? PARA FINAL
> BP
> BT
> PONCURSOR [6 3]
> ES [RESUMEN SESION IDIOMAS]
> PONCURSOR [8 7]
> ES "ACIERTOS
> PONCURSOR [20 7]
> ES "FALLOS
> PONCURSOR [10 9]
> ES :N
> PONCURSOR [22 9]
> ES :M
> PONCURSOR [8 14]
> TECLEA [TOTAL INTENTOS :]
> ES :M + :N
> PONCURSOR [10 19]
> ES [PULSA UNA TECLA]
```

Durante una pausa podemos trabajar de forma interactiva.

```
> HAZ "TT LC
> FIN
```

Y finalmente, TODO es el procedimiento principal, el que utilizamos para poner en marcha a todos los demás.

```
? PARA TODO
> BP
> BT
> OT
> HAZ "N 0
> HAZ "M 0
> LISTAS
> TO
> FINAL
> FIN
```

Cuando quieras empezar a probar qué tal está tu inglés, teclea:

```
? TODO
```

2:

En este nuevo ejercicio no aparece el procedimiento LISTAS y se incluyen los siguientes procedimientos:

El primer procedimiento que incluimos es DATOS.

Presenta la pantalla para introducir la lista de palabras inglesas y la lista de las respectivas traducciones al castellano.

```
? PARA DATOS
> HAZ "A []
> HAZ "B []
> PONCURSOR [5 3]
> ES "ENGLISH
> PONCURSOR [20 3]
> ES "CASTELLANO
> HAZ 'Y 5
> FIN
```

Con el segundo procedimiento que incluimos realizamos la lectura del teclado de las dos listas de palabras.

Para dejar de introducir datos pulsaremos RETURN en vez de la palabra en inglés y de la palabra en castellano.

```
? PARA DAT
> HAZ "Y :Y + 1
> PONCURSOR LISTA 5 :Y
> HAZ "D1 LL
> PONCURSOR LISTA 20 :Y
> HAZ "D2 LL
> SI Y :D1 = [] :D2 = [] [ALTO]
  [HAZ "A FRASE :A :D1
  HAZ "B FRASE :B :D2 HAZ "Z :Z + 1
  DAT]
> FIN
```

En el procedimiento TODO incluimos la inicialización de la variable Z que controlará el total de palabras introducidas, y la llamada a los procedimientos DATOS y DAT.

```
? PARA TODO
> BP
> BT
> OT
> HAZ "N 0
> HAZ "M 0
> DATOS
> HAZ "Z 0
> DAT
> TO
> FINAL
> FIN
```

En el procedimiento P2 cambiamos la constante 12 por la variable Z que guarda el total de palabras a gestionar.

```
? PARA P2
> HAZ "I AZAR :Z + 1
> SI :I = 0 [HAZ "I :Z - 1]
> HAZ "E ELEMENTO :I :A
> HAZ "C ELEMENTO :I :B
> PONCURSOR [8 7]
> ES :E
> PONCURSOR [4 2]
> ES [INTRODUCE LA PALABRA EN
  ESPAÑOL]
> PONCURSOR [23 7]
> HAZ "P LL
> FIN
```

Veamos un ejemplo de creacion de las dos listas:

```
? TODO
```

Una vez aparezca la siguiente pantalla:

El Logo dispone de varias primitivas para presentar texto en la pantalla.

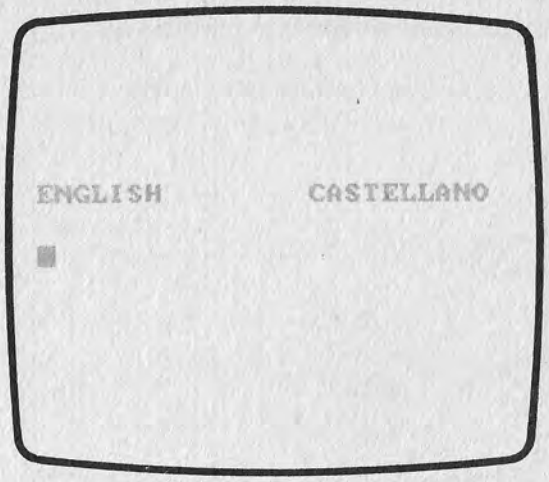


Fig. 2.

Empezaremos a introducir palabras.

ENGLISHCASTELLANO

DESK	ESCRITORIO
THIEF	LADRON
GUN	PISTOLA
HORSE	CABALLO
TICKET	BILLETE
LAWN	CESPED

Hemos creado las dos listas necesarias para continuar con el ejercicio.

En nuestros dos ejemplos hemos utilizado palabras en inglés, pero podemos escoger para trabajar cualquier otro idioma.

3:

En primer lugar, asignamos a la variable ABC el abecedario completo mediante el procedimiento P1:

```
? PARA P1
> HAZ "ABC [A B C D E F
  G H I J K L M N Ñ O P
  Q R S T U V X Y Z]
> FIN
```

P2 y P3 son los que se encargan de solicitar una palabra y de comprobar que ésta no excede de 10 caracteres, así como de comprobar que al menos contiene un elemento:

```
? PARA P2
> PONCURSOR [2 5]
> ES [INTRODUCE UNA PALABRA]
> FIN
? PARA P3
> PONCURSOR [5 10]
> PONCT FONDO
> ES "..................
> PONCT 7
> PONCURSOR [5 10]
> ES "........
> PONCURSOR [5 10]
> HAZ "P LL
> SI :P = [] [P3]
> HAZ "P ULTIPO :P
> HAZ "C CUENTA :P
> SI :C > 10 [P3]
> FIN
```

Una vez que tenemos la palabra, la escribimos, separando sus elementos dos espacios en blanco:

```
? PARA P4
> HAZ "CU ENTERO (38 - :C * 3) / 2
> HAZ "W 1
> HAZ "N 1
> REPITE :C [PONCURSOR LISTA :CU
  + :W 10 ES ELEMENTO :N ;P HAZ "N
  :N + 1 HAZ "W :W + 3]
> HAZ "I 1
> HAZ "II 1
> HAZ "III 1
> P5
> FIN
```

Al final de este procedimiento se inicializan tres variables y se hace una llamada al procedimiento P5.

```
? PARA P5
> SI :III > :C [ALTO]
> HAZ "V ELEMENTO :III :P
> SI :V = ELEMENTO :II :ABC
  [PONCURSOR LISTA :CU + :I 11
  ES :II HAZ " :I + 3
  hAZ "II 1 HAZ "III :III + 1 P5]
  [HAZ "II :II + 1 P5]
> FIN
```

P5 es el que se encarga de ir comparando los elementos de la palabra introducida con la lista del abecedario. En el momento que coinciden, se escribe el número del elemento del abecedario que se está consultando debajo del elemento correspondiente de la palabra:

```
? PARA TECLA
```

La primitiva TECLEA puede tener varios datos.

```
> PONCURSOR [14 22]
> ES [PULSA UNA TECLA]
> HAZ "T LC
> BP
> BT
> FIN
```

TECLA es el que nos pasa de pantalla una vez pulsada una tecla.

Por último, definimos TO para inicializar el proceso y englobar a todos:

```
? PARA TO
> BP
> BT
> OT
> P1
> P2
> P3
> TECLA
> P4
> FIN
```

Si lo ejecutamos:

```
? TO
```

una vez que introduzcamos una palabra, en la pantalla aparecerá:

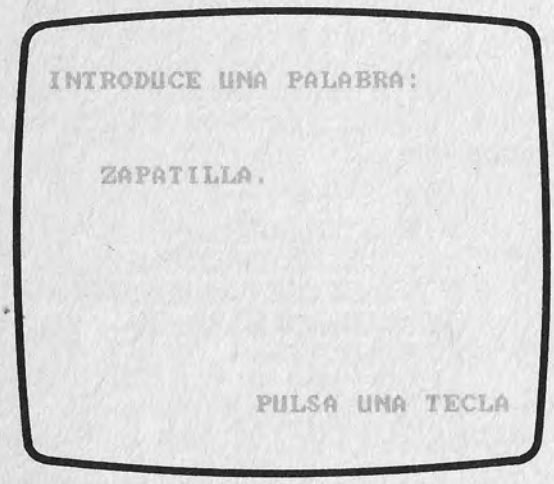


Fig. 3.

Después de pulsar una tecla tendremos:

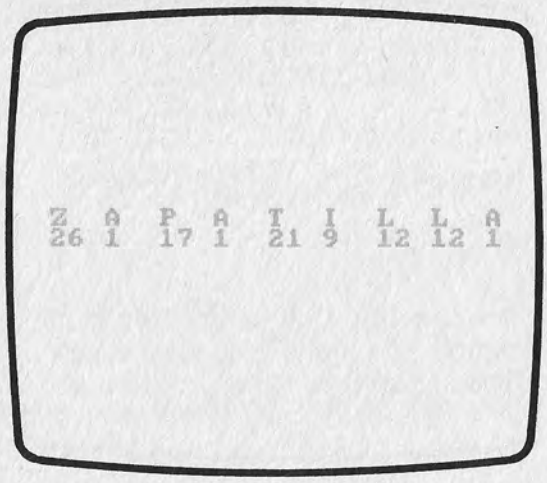


Fig. 4.

4:

El procedimiento A es el que se encarga de pedir que se introduzcan los diferentes datos que se requieren:

```
? PARA A
> PONCURSOR [2 2]
> ES [PRIMERA COLUMNA]
> PONCURSOR [19 2]
> HAZ "PC LL
> HAZ "PC UL :PC
> PONCURSOR [2 4]
> ES [ULTIMA COLUMNA]
> PONCURSOR [18 4]
> HAZ "UC LL
> HAZ "UC UL :UC
> PONCURSOR [2 6]
> ES [PRIMERA FILA]
> PONCURSOR [16 6]
> HAZ "PF LL
> HAZ "PF UL :PF
> PONCURSOR [2 8]
> ES [ULTIMA FILA]
> PONCURSOR [15 8]
> HAZ "UF LL
> HAZ "UF LL :UF
> PONCURSOR [2 10]
> ES [CARACTER DE RELLENO]
> PONCURSOR [23 10]
> HAZ "CR LL
> HAZ "CR UL :CR
> FIN
```

Cuando presentamos en la pantalla un texto con la primitiva ESCRIBE, el cursor queda posicionado en la línea siguiente.

Una vez que tenemos todos los datos y después de haber pulsado una tecla, se ejecuta el procedimiento B, que es el que calcula la zona de la pantalla que hay que rellenar y la rellena:

```
? PARA B
> HAZ "I 0
> HAZ "II 0
> REPITE (:UF - :PF) + 1
  [REPITE (:UC - :PC) + 1
  [PONCURSOR LISTA :PC + :I
  :PF + :II ES :CR HAZ "I
  :I + 1] HAZ "I 0 HAZ "II
  :II + 1]
> FIN
```

TECLA es el que se encarga de borrar la pantalla una vez que se haya pulsado una tecla.

```
? PARA TECLA
> PONCURSOR [14 22]
> ES [PULSA UNA TECLA]
> HAZ "T LC
> BP
> BT
> FIN
```

En TO englobamos todos los procedimientos definidos e inicializamos el proceso:

```
? PARA TO
> BP
> BT
> OT
> A
> TECLA
> B
> FIN
```

Para ejecutarlo, bastará con introducir:

```
? TO
```

5:

```
— ? PARA A
  > ES AZAR 1000
  > PAUSA
  > CO
  > A
  > FIN
```

INCORRECTO.

La primitiva CO no puede ir nunca dentro de un procedimiento:

```
— ? PARA B
  > REPITE 10 [PAUSA AV 10 GD 90]
  > FIN
```

CORRECTO

Nada más comenzar a ejecutarse el procedimiento habrá una pausa deteniéndose la ejecución:

```
— ? TECLEA "LA "CASA "DE "JUAN
```

INCORRECTO

Cuando la primitiva TECLEA tiene más de un dato, se debe poner todo entre paréntesis.

```
— ? PARA A1
  > HAZ "A 30
  > HAZ "B 40
  > PONCURSOR [4 10]
  > TECLEA [EL RESULTADO DE LA
    SUMA ES :]
  > ES :A + :B
  > FIN
```

CORRECTO

En la pantalla, cuando ejecutemos A1, aparecerá:

EL RESULTADO DE LA SUMA ES :70

```
— ? TECLEA ("NO "ESTA [TODO])
```

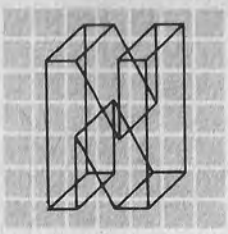
INCORRECTO

La primitiva TECLEA también debe ir dentro de los paréntesis.

Si presentamos un texto con la primitiva TECLEA, el Logo no pasa a la línea siguiente.

Si TECLEA tiene más de un dato, enciérralo todo entre paréntesis.

MANEJO DE SPRITES Y ELEMENTOS GRAFICOS



O podemos decir que un juego es bueno si no lleva un componente musical que nos lo haga más agradable. Controlar el sonido de nuestro ordenador no es nada difícil y podemos hacer que nuestro programa gane en calidad.

Ante todo, decir que hace falta conocer muy pocas cosas sobre la música, aunque alguna que otra siempre es necesaria. En el caso del AMSTRAD es importante entender una serie de conceptos, mientras que el SPECTRUM es el ordenador que más fácil resulta de programar. Por ello, empezaremos por explicar qué es un pentagrama y qué es la escala musical.

Música en nuestro ordenador

Los compositores, al componer su música, se valen de unos grafos o signos específicos. Estos signos aparecen escritos sobre lo que se llama pentagrama. El pentagrama se compone de cinco líneas horizontales y rectas, tal y como aparece en la figura 1.

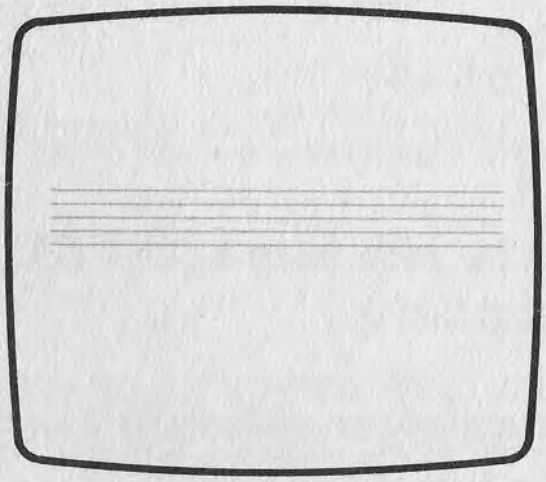


Fig. 1.

Los diferentes tonos aparecen en el pentagrama mediante unos círculos, que pueden tener un rabito o no. A estos signos se les denomina notas. Según donde coloquemos uno de estos círculos o notas en el pentagrama, representará un sonido u otro. En la figura 2 podemos ver qué notas hay y dónde hay que colocarlas.

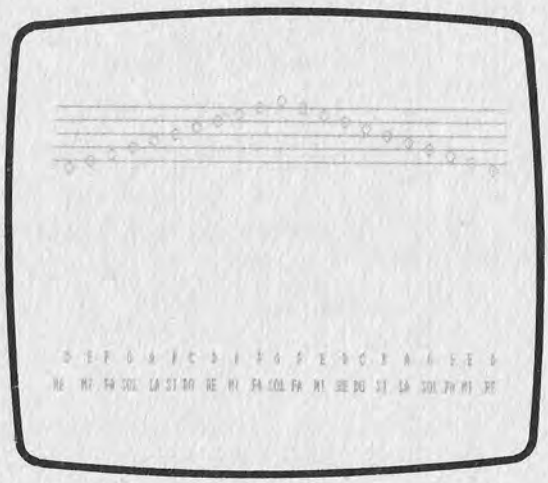


Fig. 2.

Según vemos en la figura, dichas notas son las que todos conocemos (DO, RE, MI, FA...), pero encima de cada una de ellas aparece una letra (C, D, E, F...), estas letras corresponden a la notación internacional que podemos usar en cualquier país. Esto es debido a que, mientras nosotros a la nota G la llamamos SOL, los indios (de la India) no tienen por qué llamarla igual.

Si queremos representar una nota muy alta o muy baja y no nos cabe dentro de las líneas del pentagrama, lo que podemos hacer es dibujar unos pequeños ejes encima o debajo del pentagrama para indicar la nota que queremos representar. Esto se puede entender mejor si nos fijamos en la figura 3.

Como las notas se caracterizan no sólo por la frecuencia que representan según el lu-

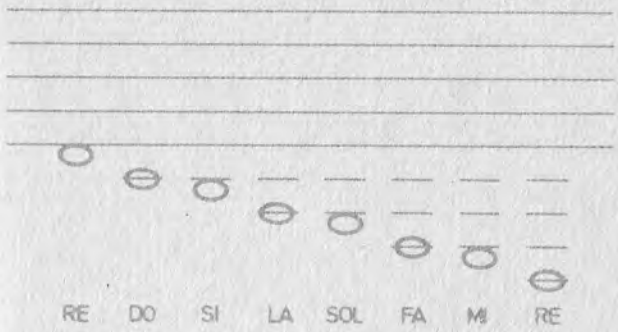


Fig. 3.

gar que ocupen, sino por la duración del sonido, tenemos que encontrar una forma de representar la duración del sonido. En la figura 4 podemos ver cómo se representan las notas según su duración y el valor que tienen.

NOTA	NOMBRE	DURACION
	REDONDA	1
	BLANCA	1/2
	NEGRA	1/4
	CORCHEA	1/8
	SEMICORCHEA	1/16
	FUSA	1/32
	SEMIFUSA	1/64

Fig. 4.

La primera nota que vemos en la tabla (REDONDA) tiene duración 1. Esto no significa nada en sí mismo, pues si variamos lo que se llama el TEMPO de la melodía, dicha unidad puede darnos una nota más larga o más corta. Por ello, los números que aparecen a la derecha de la tabla nos indican la relación que hay en la duración entre unas notas y otras.

Ya que hemos comentado algo sobre el TEMPO, hay que decir que éste aparece, normalmente, al principio de la partitura que estamos leyendo. Normalmente la forma de dar el tempo es mediante palabras italianas que nos expresan, de una forma muy colorista, cómo hay que tocar o leer la partitura. Así, nos encontramos con ritmos como LARGO, ANDANTE o VIVO. A continuación damos una tabla con los tipos y su significado. Hay que decir que cada lector de partituras interpreta el TEMPO como más le conviene, siempre dentro de unos límites.

LARGO	Muy despacio
LENTO	Muy despacio
ADAGIO	Despacio
ANDANTE	Tranquilo
ALLEGRO	De prisa
VIVACE	Muy de prisa
VIVO	Muy de prisa

Otra cosa a tener en cuenta son los silencios. No todo en una partitura son sonidos. También hay signos que nos indican que tenemos que dejar de tocar. Por supuesto, dichos silencios pueden ser o más o menos largos. La medida que se utiliza es la misma que

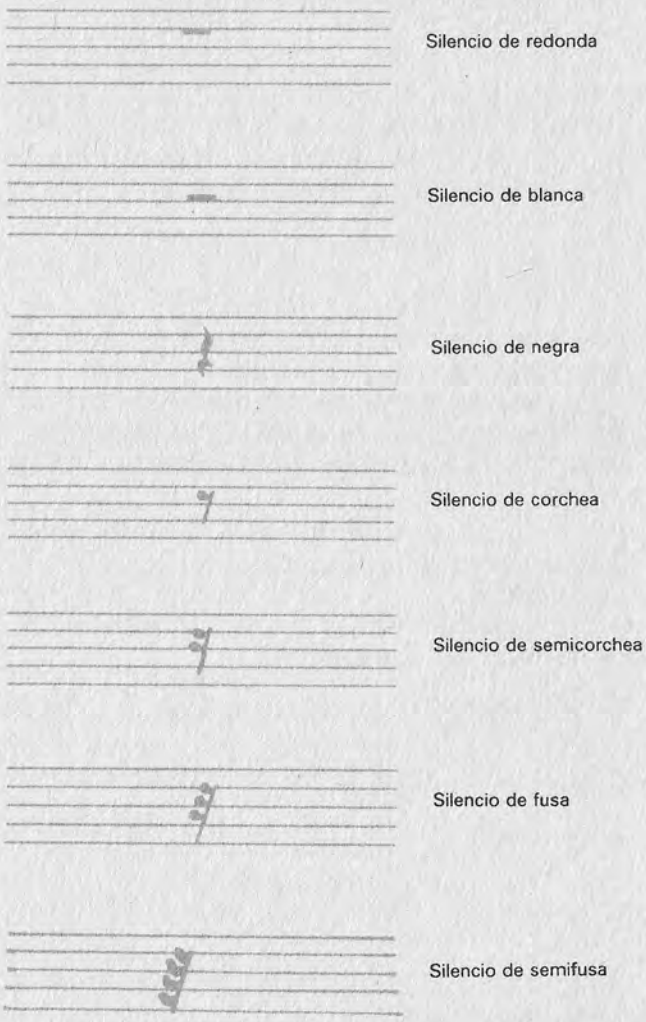


Fig. 5.

para las notas y la forma de escribirlos, en ciertos casos, nos recuerda a la nota que tiene, sonando, la misma duración. La figura 5 nos muestra una tabla con todos los silencios y su duración.

Aunque no hemos dicho todo lo que se puede decir sobre música, lo que acabamos de ver nos puede servir para entrar en este apasionante mundo. Iremos viendo cómo producir música en los distintos ordenadores y empezaremos por el SPECTRUM.

El sonido en el SPECTRUM

En el SPECTRUM sólo disponemos de una voz para producir sonidos. Aunque hay programas comerciales que, al estar escritos en código máquina, al producir música nos da la impresión de que este ordenador tuviese dos o más canales, nosotros, desde el BASIC, sólo tenemos un canal.

La instrucción que nos permite producir sonidos es el comando BEEP. BEEP va seguido de dos parámetros. El primero nos indica la duración de la nota que vamos a oír y el segundo la nota en cuestión

BEPP duración,nota

La duración ha de ir en segundos, pero podemos poner números decimales para representar tiempos menores de un segundo.

La nota ha de ser un número entre -59 y 69 que da la nota a emitir. También se pueden poner decimales para obtener cuartos y octavos de tono.

Como en el SPECTRUM las notas no se dictan con su nombre ni con la letra que internacionalmente se le asigna, vamos a ver una figura en la cual vemos la relación entre el número de nota y su nombre.

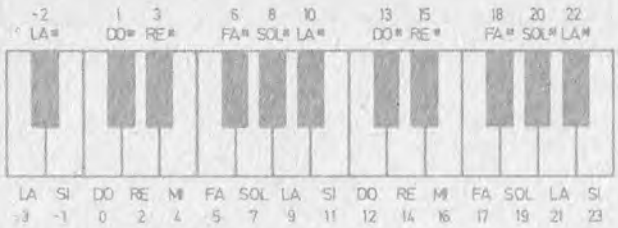


Fig. 6.

Para ver mejor cómo funcionan los dos parámetros de la sentencia BEEP, aparecen los siguientes programas:

```
100 REM ****************************************
110 REM *DURACION DE LAS NOTAS EN EL SPECTRUM *
120 REM ****************************************
130 REM
140 FOR I=0 TO 2 STEP .1
150    BEEP I,I
160 NEXT I
170 LET D=.01
180 FOR I=1 TO 10
190    PRINT AT 11,4;
200    PRINT "DURACION:    ";D;" SEGUNDOS
210    BEEP D,0
220    PAUSE 25
230    LET D=D*2
240 NEXT I
250 STOP
```

```
100 REM ********************************
110 REM * GAMA DE TONOS EN EL SPECTRUM *
120 REM ********************************
130 REM
140 FOR I=-59 TO 69
150    PRINT AT 11,11;
160    PRINT "TONO: ";I;"  ":BEEP.1,I.
170    PAUSE 25
180 NEXT I
190 STOP
```

Como en este ordenador es muy sencillo componer y tocar música, a continuación aparecen un par de programas que nos interpretan dos conocidas canciones.

```
100 REM ********************************
110 REM * ADESTE FIDELES PARA SPECTRUM *
120 REM ********************************
130 REM
140 PRINT AT 8,9;"ADESTE FIDELES"
150 PRINT AT 11,11;"VILLANCICO"
160 DIM F(8)
170 DIM N$(8,3)
180 FOR I=1 TO 8
190    READ F(I)
200    READ N$(I)
210 NEXT I
220 REM
```

```
230 REM *** LINEAS DATA CON EL NUMERO DE NOTA Y SU NOMBRE ***
240 REM
250 DATA  2,"RE"
260 DATA  4,"MI"
270 DATA  6,"FA#"
280 DATA  7,"SOL"
290 DATA  9,"LA"
300 DATA 11,"SI"
310 DATA 12,"DO"
320 DATA 14,"RE"
330 REM
340 REM *** LEER LAS DATAS CON LA MUSICA Y TOCAR CANCION ***
350 REM
360 RESTORE 470
370 FOR I=1 TO 61
380    READ D
390    READ N
400    PRINT AT 15,11;N$(N);"  "
410    BEEP D,N
420 NEXT I
430 GOTO 330
440 REM
450 REM *** LINEAS DATAS CON LA CANCION A TOCAR ***
460 REM
470 DATA 0.5,4,1,4,0.5,1,0.5,4
480 DATA 1,5,1,1,0.5,6,0.5,5,0.5,6,0.5,7
490 DATA 1,6,0.5,5,0.5,4,1,4,0.5,3,0.5,2
500 DATA 0.5,3,0.5,4,0.5,5,0.5,6,1,3,0.75,2,0.25,1
510 DATA 2,1,1,8,0.5,7,0.5,6
520 DATA 1,7,1,6,0.5,5,0.5,6,0.5,4,0.5,5
530 DATA 1,3,0.5,1,0.5,4,0.5,4,0.5,3,0.5,4,0.5,5
540 DATA 1,4,0.5,1,0.5,6,0.5,6,0.5,5,0.5,6,0.5,7
550 DATA 1,6,0.5,5,0.5,6,0.5,7,0.5,6,0.5,5,0.5,4
560 DATA 1,3,0.5,4,0.5,7,1,6,0.75,5,0.25,4,1,4
```

```
100 REM ******************************
110 REM * PARA ELISA VERSION SPECTRUM *
120 REM ******************************
130 REM
140 CLS
150 PRINT AT 9,11;"PARA ELISA"
160 PRINT AT 13,11;"BEETHOVEN"
170 FOR I=1 TO 58
180    READ A,B
190    BEEP A,B
200 NEXT I
210 DATA .2,17,.2,16
220 DATA .2,17,.2,16,.2,17,.2,12,.2,15,.2,13
230 DATA .6,10,.2,1,.2,5,.2,10
240 DATA .6,12,.2,5,.2,9,.2,12
250 DATA .6,13,.2,5,.2,17,.2,16
260 DATA .2,17,.2,16,.2,17,.2,12,.2,15,.2,13
270 DATA .6,10,.2,1,.2,5,.2,10
280 DATA .6,12,.2,5,.2,13,.2,12
290 DATA .6,10
300 DATA .2,12,.2,13,.2,15
310 DATA .6,17,.2,6,.2,18,.2,17
320 DATA .6,15,.2,6,.2,17,.2,15
330 DATA .6,13,.2,5,.2,15,.2,13
```

```
340 DATA .6,12,.2,5,.6,17
350 DATA .2,17,.6,29,.2,16
360 DATA .6,17,.2,16
370 RESTORE 210
380 FOR I=1 TO 35
390   READ A,B
400   BEEP A,B
410 NEXT I
420 RESTORE 470
430 FOR I=1 TO 102
440   READ A,B
450   BEEP A,B
460 NEXT I
470 DATA .2,13,.2,13,.2,13
480 DATA .8,13,.3,18,.1,17
490 DATA .4,17,.4,15,.3,23,.1,22
500 DATA .2,22,.2,20,.2,18,.2,17,.2,15,.2,13
510 DATA .4,11,.4,10,.1,10,.1,8,.1,10,.1,11
520 DATA .8,13,.2,15,.2,16
530 DATA .6,17,.2,17,.2,18,.2,10
540 DATA .8,13,.3,15,.1,12
550 DATA .1,13,.1,20,.1,8,.1,20,.1,10,.1,20,.1,12,.1,20,.1,13,.1,20,.1,15,.1,20
560 DATA .1,17,.1,20,.1,25,.1,24,.1,22,.1,20,.1,18,.1,17,.1,15,.1,20,.1,18,.1,15
570 DATA .1,13,.1,20,.1,8,.1,20,.1,10,.1,20,.1,12,.1,20,.1,13,.1,20,.1,15,.1,20
580 DATA .1,17,.1,20,.1,25,.1,24,.1,22,.1,20,.1,18,.1,17,.1,15,.1,20,.1,18,.1,15
590 DATA .1,17,.1,18,.1,17,.1,16,.1,17,.1,12,.1,17,.1,16,.1,17,.1,12,.1,17,.1,16
600 DATA .6,17,.2,12,.2,17,.2,16
610 DATA .6,17,.2,12,.6,17
620 DATA .2,16,.6,17,.2,16
630 RESTORE 220
640 FOR I=1 TO 56
650   READ A,B
660   BEEP A,B
670 NEXT I
680 RESTORE 210
690 FOR I=1 TO 34
700   READ A,B
710   BEEP A,B
720 NEXT I
730 RESTORE 780
740 FOR I=1 TO 77
750   READ A,B
760   BEEP A,B
770 NEXT I
780 DATA 1.2,10,1.2,14
790 DATA .8,15,.2,17,.2,18
800 DATA .8,18,.4,18
810 DATA 1.2,17
820 DATA .8,15,.2,13,.2,12
830 DATA .8,10,.4,10
840 DATA .4,10,.4,13,.4,12
850 DATA 1.2,10,1.2,14
860 DATA .8,15,.2,17,.2,18
870 DATA .8,18,.4,18
880 DATA 1.2,18
890 DATA .8,16,.2,15,.2,13
900 DATA .8,11,.4,10
910 DATA .8,9,.4,9
920 DATA 1.2,10,1.2,12
930 DATA .133,-2,.133,1,.133,5,.133,10,.133,13,.133,17,.133,15,.133,13,.133,12
940 DATA .133,10,.133,13,.133,17,.133,22,.133,25,.133,29,.133,27,.133,25,.133,24
```

```
950 DATA .133,22,.133,25,.133,29,.133,34,.133,37,.133,41,.133,39,.133,37,.133,36
960 DATA .133,35,.133,34,.133,33,.133,32,.133,31,.133,30,.133,29,.133,28,.133,27
970 DATA .133,26,.133,25,.133,24,.133,23,.133,22,.133,21,.133,20,.133,19,.133,18
980 RESTORE 220
990 FOR I=1 TO 56
1000    READ A,B
1010    BEEP A,B
1020 NEXT I
1030 RESTORE 210
1040 FOR I=1 TO 35
1050    READ A,B
1060    BEEP A,B
1070 NEXT I
1080 STOP
```

El sonido en el AMSTRAD

El AMSTRAD nos permite realizar sonidos y melodías más complicadas que el SPECTRUM. Para ello disponemos de la instrucción SOUND. Esta instrucción puede ir seguida hasta de siete parámetros, de los cuales sólo los dos primeros son imprescindibles.

SOUND C,T,D,V,EV,ET,R

donde:

C = Le dice al ordenador qué canal de sonido queremos utilizar. El AMSTRAD dispone de tres canales distintos.

T = Es el período de tono o, lo que es igual, la nota que queremos tocar.

D = Especifica la duración del sonido en centésimas de segundo (100 = 1 segundo).

V = Es el volumen, ya veremos más adelante qué es esto.

EV = Es la envolvente del volumen. Nos permitirá variar el sonido de la nota mientras se está tocando.

ET = Es la envolvente del tono. Nos permitirá variarlo mientras está sonando.

R = Es el período del ruido. Esto sirve para la generación de ruido blanco. Con este ruido podemos simular desde el choque de dos marcianos hasta el ruido de un tren.

Hagamos un estudio más detallado de cada uno de estos parámetros.

Parámetro C

Ya hemos dicho que tenemos tres canales distintos de música. Los podremos nombrar como A, B y C. Según el valor que asignemos al parámetro C, podremos hacer que suene uno u otro, incluso que suenen dos o más a la vez. Los números que representa a cada canal son los siguientes:

Canal A = 1
Canal B = 2
Canal C = 4
Sincronizar con A = 8
Sincronizar con B = 16
Sincronizar con C = 32
Retener = 64
Borrar cola = 128

Si queremos tocar alguna nota con el canal A, tendremos que hacerlo con C = 1; si fuese con el canal B, entonces C = 2. Si queremos tocar con el canal A mientras está sonando el C y queremos sincronizar ambos canales, entonces tendremos que poner C = 8 + 4. El ocho (8) significa que queremos que suene el canal A, con la nota que se da a continuación y que queremos que el sonido quede sincronizado con el canal C, cuya nota dimos anteriormente.

Si C tiene valor 64, entonces retenemos el sonido durante el tiempo que queremos. Esto significa que dejará de sonar durante un cierto tiempo. Para que el sonido continúe, tendremos que poner:

RELEASE C

donde C es el canal o canales que queremos liberar. C puede tener un valor entre 1 y 7.

1 = Canal A
2 = Canal B
3 = Canales A y B
4 = Canal C
5 = Canales A y C
6 = Canales B y C
7 = Canales A, B y C

Si el valor de C contiene a 128, entonces se borra toda la cola de música que hubiese retenida.

El parámetro T

Como dijimos, este parámetro le dice al ordenador qué nota tiene que tocar. Para ello podemos poner un número entre 0 (agudo) y

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	32.703	3822	-0.007%	
C# DO#	34.648	3608	+0.007%	
D RE	36.708	3405	-0.007%	
D# RE#	38.891	3214	-0.004%	
E MI	41.203	3034	+0.009%	
F FA	43.654	2863	-0.016%	
F# FA#	46.249	2703	+0.009%	Octava -3
G SOL	48.999	2551	-0.002%	
G# SOL#	51.913	2408	+0.005%	
A LA	55.000	2273	+0.012%	
A# LA#	58.270	2145	-0.008%	
B SI	61.735	2025	+0.011%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	65.406	1911	-0.007%	
C# DO#	69.296	1804	+0.007%	
D RE	73.416	1703	+0.022%	
D# RE#	77.782	1607	-0.004%	
E MI	82.407	1517	+0.009%	
F FA	87.307	1432	+0.019%	
F# FA#	92.499	1351	-0.028%	Octava -2
G SOL	97.999	1276	+0.037%	
G# SOL#	103.826	1204	+0.005%	
A LA	110.000	1136	-0.032%	
A# LA#	116.541	1073	+0.039%	
B SI	123.471	1012	-0.038%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	130.813	956	+0.046%	
C# DO#	138.591	902	+0.007%	
D RE	146.832	851	-0.037%	
D# RE#	155.564	804	+0.058%	
E MI	164.814	758	-0.057%	
F FA	174.614	716	+0.019%	
F# FA#	184.997	676	+0.046%	Octava -1
G SOL	195.998	638	+0.037%	
G# SOL#	207.652	602	+0.005%	
A LA	220.000	568	-0.032%	
A# LA#	233.082	536	-0.055%	
B SI	246.942	506	-0.038%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	261.626	478	+0.046%	DO media
C# DO#	277.183	451	+0.007%	
D RE	293.665	426	+0.081%	
D# RE#	311.127	402	+0.058%	
E MI	329.628	379	-0.057%	
F FA	349.228	358	+0.019%	
F# FA#	369.994	338	+0.046%	Octava 0
G SOL	391.995	319	+0.037%	
G# SOL#	415.305	301	+0.005%	
A LA	440.000	284	-0.032%	LA internacional
A# LA#	466.164	268	-0.055%	
B SI	493.883	253	-0.038%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	523.251	239	+0.046%	
C# DO#	554.365	225	-0.215%	
D RE	587.330	213	+0.081%	
D# RE#	622.254	201	+0.058%	
E MI	659.255	190	+0.206%	
F FA	698.457	179	+0.019%	
F# FA#	739.989	169	+0.046%	Octava 1
G SOL	783.991	159	-0.277%	
G# SOL#	830.609	150	-0.328%	
A LA	880.000	142	-0.032%	
A# LA#	932.328	134	-0.055%	
B SI	987.767	127	+0.356%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	1046.502	119	-0.394%	
C# DO#	1108.731	113	+0.229%	
D RE	1174.659	106	-0.390%	
D# RE#	1244.508	100	-0.441%	
E MI	1318.510	95	+0.206%	
F FA	1396.913	89	-0.543%	
F# FA#	1479.978	84	-0.548%	Octava 2
G SOL	1567.982	80	+0.350%	
G# SOL#	1661.219	75	-0.328%	
A LA	1760.000	71	-0.032%	
A# LA#	1864.655	67	-0.055%	
B SI	1975.533	63	-0.435%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	2093.004	60	+0.462%	
C# DO#	2217.461	56	-0.662%	
D RE	2349.318	53	-0.390%	
D# RE#	2489.016	50	-0.441%	
E MI	2637.021	47	-0.855%	
F FA	2793.826	45	+0.574%	
F# FA#	2959.955	42	-0.548%	Octava 3
G SOL	3135.963	40	+0.450%	
G# SOL#	3322.438	38	+0.992%	
A LA	3520.000	36	+1.357%	
A# LA#	3729.310	34	+1.417%	
B SI	3951.066	32	+1.134%	

NOTA	FRECUENCIA	PERIODO	ERROR RELATIVO	
C DO	4186.009	30	+0.462%	
C# DO#	4434.922	28	-0.662%	
D RE	4698.636	27	+1.469%	
D# RE#	4978.032	25	-0.441%	
E MI	5274.041	24	+1.246%	
F FA	5587.652	22	-1.685%	
F# FA#	5919.911	21	-0.548%	Octava 4
G SOL	6271.927	20	+0.350%	
G# SOL#	6644.875	19	+0.992%	
A LA	7040.000	18	+1.357%	
A# LA#	7458.621	17	+1.417%	
B SI	7902.133	16	+1.134%	

Fig. 7.

```
100 REM ***********************************
110 REM * ESCALA CROMATICA EN EL AMSTRAD *
120 REM ***********************************
130 REM
140 CLS
150 FOR I=1 TO 12
160     READ T
170     READ N$
180     LOCATE 19,13
190     PRINT N$;"  "
200     SOUND 1,T
210     FOR J=1 TO 500
220     NEXT J
230 NEXT I
240 DATA 478,"DO"
250 DATA 451,"DO#"
260 DATA 426,"RE"
270 DATA 402,"RE#"
280 DATA 379,"MI"
290 DATA 358,"FA"
300 DATA 338,"FA#"
310 DATA 319,"SOL"
320 DATA 301,"SOL#"
330 DATA 284,"LA"
340 DATA 268,"LA#"
350 DATA 253,"SI"
```

4095 (grave). En la figura 7 podemos ver las notas, la frecuencia de cada nota y el período o número que tenemos que poner para reproducir dicha nota.

También podemos ver en la figura 7 el error relativo que se produce al dar una nota con respecto a la nota real. Normalmente, al ser muy pequeño el error, el oído humano no puede captar la diferencia.

El parámetro D

Nos indica la duración del sonido en centésimas de segundo. Puede tener un valor

entre -32768 y 32767. Si dicho valor es igual a cero (0), entonces la duración del sonido irá controlada por la envolvente. Si es menor que cero (>0), entonces le dice al ordenador el número de veces que se repite la envolvente. Normalmente tiene un valor implícito de 20.

```
100 REM ************************************
110 REM * ESCALA Y DURACIONES EN EL AMSTRAD *
120 REM ************************************
130 REM
140 CLS
150 LET D=2
160 FOR I=1 TO 8
170    READ T
180    READ N$
190    LOCATE 16,10
200    PRINT "NOTA: ";N$;"  "
210    LOCATE 10,16
220    PRINT "DURACION ";D/100;" SEGUNDOS   "
230    SOUND 1,T,D
240    LET D=D*2
250    FOR J=1 TO 500
260    NEXT J
270 NEXT I
280 DATA 478,"DO"
290 DATA 426,"RE"
300 DATA 379,"MI"
310 DATA 358,"FA"
320 DATA 319,"SOL"
330 DATA 284,"LA"
340 DATA 253,"SI"
350 DATA 239,"DO"
```

El parámetro V

Nos permite dar un volumen distinto a cada sonido que produzcamos. Puede tomar valores entre 0 y 15, siendo 0 el mínimo y 15 el máximo. Si no se especifica ningún valor, el ordenador lo toma como volumen 12.

```
100 REM ***********************************
110 REM * ESCALAS Y VOLUMEN EN EL AMSTRAD *
120 REM ***********************************
130 REM
140 CLS
150 FOR I=1 TO 15 STEP 2
160    GOSUB 220
170 NEXT I
180 FOR I=15 TO 1 STEP -2
190    GOSUB 220
200 NEXT I
210 END
220 REM
230 REM *** RUTINA DE SOUND ***
240 REM
250 READ T
260 READ N$
270 LOCATE 16,10
280 PRINT "NOTA: ";N$;"  "
290 LOCATE 16,16
300 PRINT "VOLUMEN: ";I;"  "
310 SOUND 1,T,50,I
320 FOR J=1 TO 500
```

```
330 NEXT J
340 RETURN
350 REM
360 REM *** LINEAS DE DATA ***
370 REM
380 DATA 239,"DO",213,"RE"
390 DATA 190,"MI",179,"FA"
400 DATA 159,"SOL",142,"LA"
410 DATA 127,"SI",119,"DO"
420 DATA 119,"DO",127,"SI"
430 DATA 142,"LA",159,"SOL"
440 DATA 179,"FA",190,"MI"
450 DATA 213,"RE",239,"DO"
```

El parámetro EV

Con este parámetro le decimos al ordenador que queremos utilizar la envolvente de volumen EV. EV deberá ser un número entre 1 y 15 y, antes de incluir este parámetro tenemos que haber definido la envolvente a utilizar.

Para definir una envolvente utilizaremos la instrucción ENV, cuya sintaxis es:

ENV N,E1,A1,D1,E2,A2,D2,E3,A3,
D3,E4,A4,D4,E5,A5,D5

donde:

N = Es el número de envolvente que estamos definiendo y que es el número de referencia que utilizaremos en la sentencia SOUND.

En = Indica el número de escalones en cada sección del sonido. Una sección es una parte del sonido donde éste disminuye o aumenta. Como tenemos desde E1 hasta E5, esto significa que podemos tener hasta 5 secciones distintas. Según esto, este parámetro le indica al ordenador cuántas etapas de volumen queremos que evolucione la nota antes de concluir. Puede tomar valores entre 0 y 127.

An = Define la altura de cada escalón de forma que el nivel de volumen de cada escalón puede variar; con respecto al anterior, entre 1 y 15; sin embargo, los parámetros aceptados pueden ir desde -128 a +127.

Dn = Indica la duración de cada escalón en centésimas de segundo. Puede tomar valores entre 0 y 255.

El programa 8 nos muestra un sonido con envolvente de volumen definida en cinco secciones. La figura 8 nos muestra la envolvente que utilizamos en este programa.

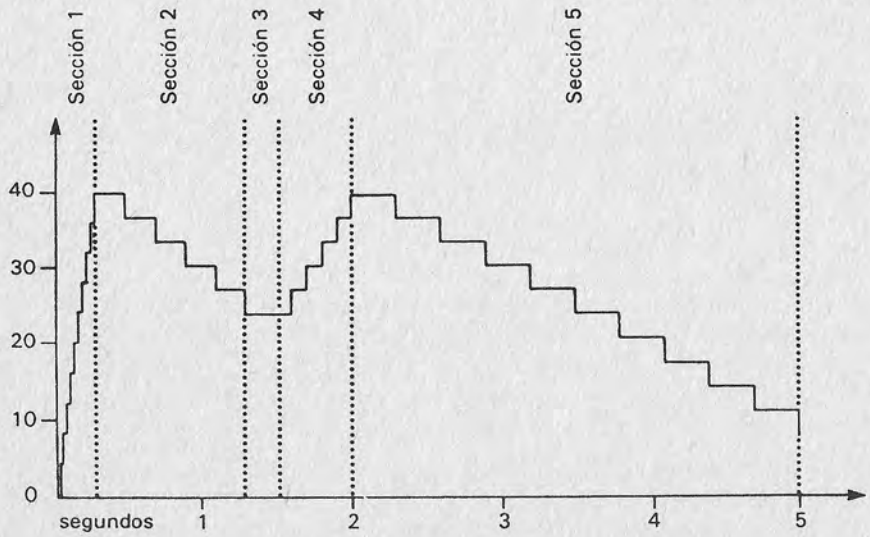


Fig. 8.

```
100 REM ***************************************************
110 REM * EJEMPLO DE ENVOLVENTE DE VOLUMEN EN EL AMSTRAD *
120 REM ***************************************************
130 REM
140 CLS
150 LOCATE 13,13
160 PRINT "EJEMPLO DEL COMANDO ENV"
170 ENV 1,10,4,3,5,-3,20,1,0,20,5,3,10,10,-3,30
180 SOUND 1,478,0,0,1,0,0
190 GOTO 180
```

El parámetro ET

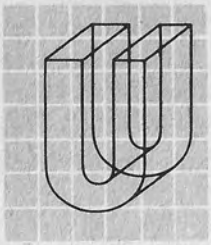
Al igual que con EV, ET le dice al ordenador que queremos utilizar una envolvente de tono que nosotros mismos hemos definido. Para definirla, utilizaremos la instrucción:

ENT N,E1,A1,D1,E2,A2,D2,E3,A3,D3,E4, A4, D4,E5,A5,D5

Estos parámetros tienen el mismo significado que en ENV, salvo que en vez de actuar sobre el volumen actúan sobre el tono de la nota que va a sonar. Por otro lado, el argumento EN puede tomar valores entre 0 y 239.

El programa 9 nos muestra un ejemplo de su utilización.

```
100 REM ******************************
110 REM * EJEPLO DE ENT EN EL AMSTRAD *
120 REM ******************************
130 REM
140 CLS
150 LOCATE 13,13
160 PRINT "EJEMPLO DEL COMADO ENT"
170 ENT 1,65,5,1,10,-2,10,10,2,5,30,-5,1
180 SOUND 1,478,50,15,0,1,0
190 GOTO 180
```



NA de las aplicaciones que más sacan partido a la alta resolución es la representación gráfica de funciones matemáticas. Se puede decir que la alta resolución nació de la necesidad de poder representar dichas funciones con cierta calidad.

El programa que vamos a ver a continuación nos permitirá representar cualquier tipo de función por complicada que ésta sea.

Con este programa incluso podremos dibujar las asíntotas, hacer ZOOM de una parte de la función, etc.

```
10 REM ******************************
20 REM ***       GRAFICAS!        ***
30 REM *** Un programa realizado ***
40 REM ***          Por           ***
50 REM *** Carlos A. Maria Morin ***
60 REM ***                        ***
70 REM ***     (C) Ediciones      ***
80 REM ***  Siglo Cultural-1987   ***
90 REM ******************************
100 REM
110 REM ********************************
120 REM ******** INICIALIZACION ********
130 REM ********************************
140 REM
150 BORDER 1
160 PAPER 0
170 PEN 5
180 CLS
190 MODE 0
200 LOCATE 3,12
210 PRINT"G R A F I C O S"
220 FOR ret=1 TO 3000
230 NEXT
240 MODE 2
250 DIM aumento$(3)
260 A1=1
270 A2=1
280 X5=0
290 aumento=1
300 k1$(1)="T"
310 k1$(2)="S"
320 K1$(3)="S"
330 k1$(4)="S"
340 k1$(5)="S"
350 k1$(6)="G"
360 k1$(7)="N"
370 k1$(8)="N"
380 SYMBOL 244,0,0,60,66,129,129,66,60
```

```
390 SYMBOL 245,0,0,60,66,129,129,66,60
400 WINDOW #1,1,40,25,25
410 KEY 156,CHR$(13)+"goto 380"+CHR$(13)
420 KEY 157,"cls:list"+CHR$(13)
430 KEY DEF 15,0,48,156,157
440 REM
450 REM ******************************
460 REM ** PROGRAMA PRINCIPAL 'MENU' **
470 REM ******************************
480 REM
490 CLS
500 BORDER 24
510 CLS
520 PRINT CHR$(22);CHR$(1)
530 LOCATE 32,2
540 PRINT" G R A F I C O S "
550 LOCATE 32,2
560 PRINT" _______________ "
570 PRINT CHR$(22);CHR$(0)
580 PRINT:PRINT:PRINT
590 DEF FNy(x)=((x*(SQR(1-x)))/(SQR(x+1)))
600 PRINT:PRINT:PRINT:PRINT:PRINT
610 PRINT SPACE$(29);"1...Dibujar grafica."
620 PRINT SPACE$(29);"2...Escribir formula."
630 PRINT SPACE$(29);"3...Definir formato."
640 PRINT SPACE$(29);"4...Salir del programa."
650 PRINT:PRINT:PRINT
660 PRINT:PRINT:PRINT
670 LOCATE 31,24
680 PRINT " <<ELIGE OPCION>>  "
690 k$=INKEY$
700 IF k$="" THEN 690 ELSE k=ASC(k$)-48
710 IF k<1 OR k>4 THEN SOUND 1,60,10,15 ELSE 780
720 LOCATE 31,24
730 PRINT "<<OPCION INVALIDA>>"
740 CLEAR INPUT
750 FOR ret=1 TO 1500
760 NEXT
770 GOTO 670
780 ON k GOSUB 1120,1040,840,800
790 GOTO 490
800 CLS
810 BORDER 1
820 END
830 REM
840 REM ******************************
850 REM **** SELECCION DEL FORMATO ****
860 REM ******************************
870 REM
880 CLS
890 PRINT:PRINT"FORMATO:"
900 PRINT:PRINT:RESTORE
910 FOR f=1 TO 8
920 READ af$,neg$,men$
930 PRINT men$;
940 IF f=7 THEN PRINT "Margen cuando XY "CHR$(154)+CHR$(243)" "CHR$(244)+CHR$(24
5)".    (";
950 PRINT k1$(f)") ";
960 k$=INKEY$
970 IF k$="" THEN 960 ELSE k$=UPPER$(k$)
980 IF k$<>"" THEN IF k$=af$ OR k$=neg$ THEN k1$(f)=k$
990 IF k$<>af$ AND k$<>neg$ THEN PRINT ELSE PRINT k1$(f)
1000 NEXT f
1010 RETURN
1020 STOP
1030 REM
```

```
1040 REM ======== VOLVER AL MENU ========
1050 REM
1060 CLS
1070 LOCATE 21,24
1080 PRINT"Cambiar la ecuacion y pulsar  [SHIFT] y [0]"
1090 LOCATE 20,10
1100 EDIT 590
1110 GOTO 490
1120 CLS
1130 PAPER 2
1140 PEN 5
1150 BORDER 1
1160 CLS
1170 IF k1$(6)="R" THEN RAD ELSE DEG
1180 REM
1190 REM ********************************
1200 REM ********** DIBUJA EJES **********
1210 REM ********************************
1220 REM
1230 ORIGIN 440,200
1240 MOVE -200,0:DRAW -200,200
1250 MOVE -200,199:DRAW 200,199
1260 MOVE 199,199:DRAW 200,-200
1270 MOVE 0,0:MOVE -200,0
1280 DRAW -200,-199:MOVE -200,-200
1290 DRAW 200,-200:MOVE 199,-200
1300 DRAW 199,200:MOVE 5,38
1310 DRAW -5,38:MOVE 5,76
1320 DRAW -5,76:MOVE 5,114
1330 DRAW -5,114
1340 IF k1$(2)="S" THEN MOVE 200,0:DRAW -200,0
1350 MOVE 5,152:DRAW -5,152
1360 MOVE 5,190:DRAW -5,190
1370 MOVE 5,-38 :DRAW -5,-38
1380 MOVE 5,-76:DRAW -5,-76
1390 MOVE 5,-114:DRAW -5,-114
1400 MOVE 5,-152:DRAW -5,-152
1410 MOVE 5,-190:DRAW -5,-190
1420 PLOT 0,0
1430 IF k1$(3)="S" THEN MOVE 0,200:DRAW 0,-200
1440 MOVE 38,5:DRAW 38,-5
1450 MOVE 76,5:DRAW 76,-5
1460 MOVE 114,5:DRAW 114,-5
1470 MOVE 152,5:DRAW 152,-5
1480 MOVE 190,5:DRAW 190,-5
1490 MOVE -38,5:DRAW -38,-5
1500 MOVE -76,5:DRAW -76,-5
1510 MOVE -114,5:DRAW -114,-5
1520 MOVE -152,5:DRAW -152,-5
1530 MOVE -190,5:DRAW -190,-5
1540 IF k1$(8)="S" THEN GOSUB 3430
1550 REM
1560 REM ********************************
1570 REM * SELECCION DE ESCALA Y AUMENTO *
1580 REM ********************************
1590 REM
1600 LOCATE 1,1
1610 PRINT "Escala del eje X:";A1;
1620 INPUT ;A1$
1630 IF A1$<>"" THEN A1=VAL(A1$)
1640 LOCATE 18,1
1650 PRINT A1;"      "
1660 LOCATE 1,2
1670 PRINT"Escala del eje Y:";A2;
1680 INPUT A2$
1690 IF A2$<>"" THEN A2=VAL(A2$)
```

```
1700 LOCATE 18,2
1710 PRINT A2;"     "
1720 LOCATE 31,12
1730 PRINT"X=";INT(1000*200/A1)/1000
1740 LOCATE 58,1
1750 PRINT"Y=";INT(1000*200/A2)/1000
1760 LOCATE 1,3
1770 PRINT"Ampliacion en eje X (0-9)";
1780 INPUT aumento$(1)
1790 LOCATE 27,3
1800 PRINT " ";aumento$(1)
1810 IF aumento$(1)<>"" AND aumento$(1)<>"0" THEN 1820 ELSE 1870
1820 IF aumento$(1)<>"1" AND aumento$(1)<>"2" THEN 1830 ELSE 1870
1830 IF aumento$(1)<>"3" AND aumento$(1)<>"4" THEN 1840 ELSE 1870
1840 IF aumento$(1)<>"5" AND aumento$(1)<>"6" THEN 1850 ELSE 1870
1850 IF aumento$(1)<>"7" AND aumento$(1)<>"8" THEN 1860 ELSE 1870
1860 IF aumento$(1)<>"9" THEN 1760 ELSE 1870
1870 LOCATE 1,4
1880 PRINT"Ampliacion en eje Y (0-9)";
1890 INPUT aumento$(2)
1900 LOCATE 27,4
1910 PRINT " ";aumento$(2)
1920 IF aumento$(2)<>"" AND aumento$(2)<>"0" THEN 1930 ELSE 1980
1930 IF aumento$(2)<>"1" AND aumento$(2)<>"2" THEN 1940 ELSE 1980
1940 IF aumento$(2)<>"3" AND aumento$(2)<>"4" THEN 1950 ELSE 1980
1950 IF aumento$(2)<>"5" AND aumento$(2)<>"6" THEN 1960 ELSE 1980
1960 IF aumento$(2)<>"7" AND aumento$(2)<>"8" THEN 1970 ELSE 1980
1970 IF aumento$(2)<>"9" THEN 1870 ELSE 1980
1980 IF aumento$(1)="" THEN aumento(1)=1
1990 IF aumento$(1)="0" THEN aumento(1)=10
2000 IF aumento$(1)="1" THEN aumento(1)=20
2010 IF aumento$(1)="2" THEN aumento(1)=30
2020 IF aumento$(1)="3" THEN aumento(1)=40
2030 IF aumento$(1)="4" THEN aumento(1)=50
2040 IF aumento$(1)="5" THEN aumento(1)=60
2050 IF aumento$(1)="6" THEN aumento(1)=70
2060 IF aumento$(1)="7" THEN aumento(1)=80
2070 IF aumento$(1)="8" THEN aumento(1)=90
2080 IF aumento$(1)="9" THEN aumento(1)=100
2090 IF aumento$(2)="" THEN aumento(2)=1
2100 IF aumento$(2)="0" THEN aumento(2)=10
2110 IF aumento$(2)="1" THEN aumento(2)=20
2120 IF aumento$(2)="2" THEN aumento(2)=30
2130 IF aumento$(2)="3" THEN aumento(2)=40
2140 IF aumento$(2)="4" THEN aumento(2)=50
2150 IF aumento$(2)="5" THEN aumento(2)=60
2160 IF aumento$(2)="6" THEN aumento(2)=70
2170 IF aumento$(2)="7" THEN aumento(2)=80
2180 IF aumento$(2)="8" THEN aumento(2)=90
2190 IF aumento$(2)="9" THEN aumento(2)=100
2200 IF k1$(4)="N" THEN LOCATE 1,5:INPUT "STEP: ";S
2210 IF k1$(4)="N" THEN ELSE S=2/A1:LOCATE 1,5:PRINT"STEP:";"Autom.";S;"  "
2220 IF S=0 THEN S=2/A1
2230 IF k1$(1)="T" THEN 2820 ELSE 2260
2240 END
2250 REM
2260 REM ************************************
2270 REM * TRATAMIENTO GRAFICO DE FUNCIONES *
2280 REM ************************************
2290 REM
2300 ON ERROR GOTO 3380
2310 LOCATE 1,8
2320 PRINT"Funcion:f(x)"
2330 MOVE X5,0
2340 FOR X=X5 TO 200/A1 STEP S
```

```
2350 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2360 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 2410
2370 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 2
420
2380 px=X*A1*aumento(1)
2390 py=INT(FN y(X)*A2*aumento(2))
2400 PLOT px,py
2410 NEXT
2420 LOCATE 1,8
2430 PRINT"Funcion:f(-x)"
2440 MOVE -200/A1,0
2450 FOR X=(-200/A1)/aumento(2) TO X5 STEP S
2460 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2470 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 2520
2480 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 2
530
2490 px=X*A1*aumento(1)
2500 py=INT(FN y(X)*A2*aumento(2))
2510 PLOT px,py
2520 NEXT
2530 LOCATE 1,8
2540 PRINT"Funcion:-f(x)"
2550 MOVE X5,0
2560 FOR X=X5 TO 200/A1 STEP S
2570 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2580 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 2630
2590 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 2
640
2600 px=X*A1*aumento(1)
2610 py=INT(-(FN y(X)*A2*aumento(2)))
2620 PLOT px,py
2630 NEXT
2640 LOCATE 1,8
2650 PRINT"Funcion:-f(-x)"
2660 MOVE -200/A1,0
2670 FOR X=((-200/A1)/aumento(2)) TO X5 STEP S
2680 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2690 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 2740
2700 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 2
750
2710 px=X*A1*aumento(1)
2720 py=INT(-(FN y(X)*A2*aumento(2)))
2730 PLOT px,py
2740 NEXT
2750 LOCATE 1,12
2760 PRINT"*****************"
2770 LOCATE 1,13
2780 PRINT"*PULSA UNA TECLA*"
2790 LOCATE 1,14:PRINT"*****************"
2800 CALL &BB06:REM EN ESPERA DE QUE SE PULSE UNA TECLA
2810 GOTO 490
2820 ON ERROR GOTO 3380
2830 LOCATE 1,8
2840 PRINT"Funcion:f(x)"
2850 MOVE X5,0
2860 FOR X=X5 TO 200/A1 STEP S
2870 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2880 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 2930
2890 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN
2940
2900 px=X*A1*aumento(1)
```

```
2910 py=INT(FN y(X)*A2*aumento(2))
2920 DRAW px,py
2930 NEXT
2940 LOCATE 1,8
2950 PRINT"Funcion:f(-x)"
2960 MOVE -200,0
2970 FOR X=(-200/A1)/aumento(2) TO X5 STEP S
2980 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
2990 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 3040
3000 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 3
050
3010 px=X*A1*aumento(1)
3020 py=INT(FN y(X)*A2*aumento(2))
3030 DRAW px,py
3040 NEXT
3050 LOCATE 1,8
3060 PRINT"Funcion:-f(x)"
3070 MOVE X5,0
3080 FOR X=X5 TO 200/A1 STEP S
3090 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
3100 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 3150
3110 IF k1$(7)="N" THEN IF XPOS>205 OR XPOS<-205 OR YPOS>205 OR YPOS<-205 THEN 3
160
3120 px=X*A1*aumento(1)
3130 py=INT(-(FN y(X)*A2*aumento(2)))
3140 DRAW px,py
3150 NEXT
3160 LOCATE 1,8
3170 PRINT"Funcion:-f(-x)"
3180 MOVE -200,0
3190 FOR X=(-200/A1)/aumento(2) TO X5 STEP S
3200 IF k1$(5)="S" THEN LOCATE 1,6:PRINT "X:";XPOS;:LOCATE 1,7:PRINT"Y:";YPOS"
     "
3210 IF X*A1>32750 OR Y*A2>32750 OR X*A1<-32750 OR Y*A2<-32750 THEN 3260
3220 IF k1$(7)="N" THEN IF X>205 OR X<-205 OR YPOS>205 OR YPOS<-205 THEN 3270
3230 px=X*A1*aumento(1)
3240 py=INT(-(FN y(X)*A2*aumento(2)))
3250 DRAW px,py
3260 NEXT
3270 LOCATE 1,12
3280 PRINT"*****************"
3290 LOCATE 1,13
3300 PRINT"*PULSA UNA TECLA*"
3310 LOCATE 1,14
3320 PRINT"*****************"
3330 CALL &BB06:REM EN ESPERA DE QUE SE PULSE UNA TECLA
3340 GOTO 490
3350 REM
3360 REM ===== TRATAMIENTO DE ERRORES =====
3370 REM
3380 LOCATE 1,9
3390 PRINT "***ERROR SOBREPASA RANGO"
3400 RESUME NEXT
3410 END
3420 REM
3430 REM ********************************
3440 REM ******* DIBUJA CUADRICULA *******
3450 REM ********************************
3460 REM
3470 ORIGIN 440,200
3480 MOVE 200,38:DRAW -200,38
3490 MOVE 200,76:DRAW -200,76
3500 MOVE 200,114:DRAW -200,114
3510 MOVE 200,152:DRAW -200,152
```

```
3520 MOVE 200,190:DRAW -200,190
3530 MOVE 200,-38 :DRAW -200,-38
3540 MOVE 200,-76:DRAW -200,-76
3550 MOVE 200,-114:DRAW -200,-114
3560 MOVE 200,-152:DRAW -200,-152
3570 MOVE 200,-190:DRAW -200,-190
3580 MOVE 38,200:DRAW 38,-200
3590 MOVE 76,200:DRAW 76,-200
3600 MOVE 114,200:DRAW 114,-200
3610 MOVE 152,200:DRAW 152,-200
3620 MOVE 190,200:DRAW 190,-200
3630 MOVE -38,200:DRAW -38,-200
3640 MOVE -76,200:DRAW -76,-200
3650 MOVE -114,200:DRAW -114,-200
3660 MOVE -152,200:DRAW -152,-200
3670 MOVE -190,200:DRAW -190,-200
3680 RETURN
3690 END
3700 REM
3710 REM ********************************
3720 REM ************ DATAS **************
3730 REM ********************************
3740 REM
3750 DATA P,T,1...Punto o Trazo.(
3760 DATA S,N,2...Dibuja eje x. (
3770 DATA S,N,3...Dibuja eje y. (
3780 DATA S,N,4...STEP   Auto.  (
3790 DATA S,N,5...Escribe valores de x e y.  (
3800 DATA R,G,6...Fn. Tri. Radianes o Grados.(
3810 DATA S,N,7...
3820 DATA S,N,8...Dibuja cuadricula.         (
```

El programa parece realmente indicado para aquellos estudiantes que han alcanzado un nivel igual o superior a tercero de BUP. El programa les podrá servir para comprobar que sus ejercicios están bien hechos. Este programa también tiene otra utilidad. Esta consiste en poder enseñar a aquellos a los que las matemáticas no terminan de convencerles, que son una ciencia bonita, que no todo son números y letras, que toda esa avalancha de números, si los ordenamos adecuadamente, puede formar dibujos realmente increíbles.

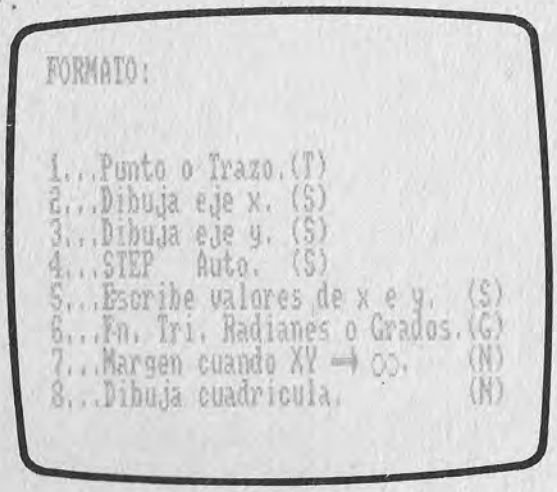


Fig. 1. Podemos variar el formato de salida.

Al ejecutar el programa nos aparecerá un menú con cuatro opciones. Estas son:

1. Dibujar gráfica.
2. Escribir fórmula.
3. Definir formato.
4. Salir del programa.

Si pulsamos la opción 2, nos aparecerá en pantalla la función que en ese momento tiene almacenado el ordenador. Para cambiarlo sólo tenemos que teclear la nueva función y pulsar las teclas SHIFT y 0 a la vez.

Con la opción 3 podemos definir el formato que tendrá la representación gráfica de dicha función en la pantalla. Podremos decir si queremos que aparezcan los ejes de coordenadas, si queremos dibujar con puntos o líneas, si queremos asíntotas, cuadrículas, etcétera.

Con la opción 1 representamos la función que hemos definido con la opción 2 y cuyas características hemos dado con la opción 3. Una vez que el ordenador haya dibujado la ventana donde dibujará dicha función, nos preguntará qué escala queremos que tenga el eje X y el eje Y. Ambos pueden tener una gra-

duación distinta. Por otro lado, también podemos decir si queremos que haya ampliación en X o en Y o en ambas a la vez. Esto nos puede servir para que podamos ver con más claridad una parte de la funcion que no se ve bien.

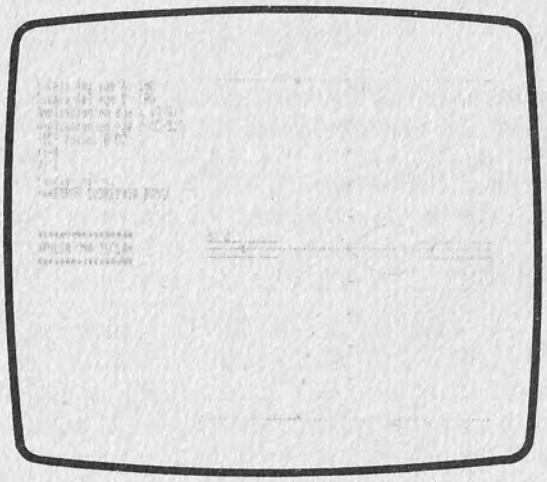


Fig. 2.

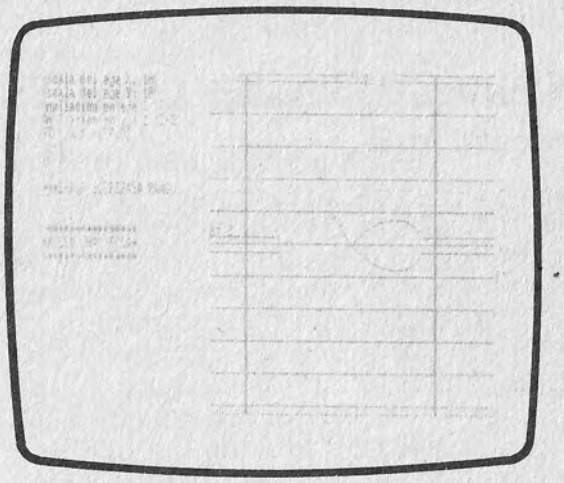
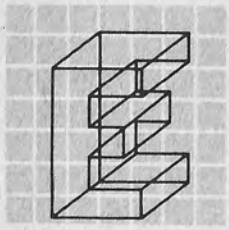


Fig. 3. Representación con cuadrícula.

En el caso de que no hayamos querido el STEP automático cuando hemos definido el formato, el ordenador nos preguntará qué STEP deseamos. Como norma general, el STEP automático será suficiente para nuestras necesidades.

Tarjeta digitalizadora de vídeo. Segunda parte



En el tomo anterior se ha descrito el funcionamiento en general de la tarjeta de digitalización de imágenes de vídeo y el circuito de tratamiento analógico de la señal. Continuamos con la descripción de los circuitos de control digital y del programa que permite su empleo a través de un ordenador personal, con interfaz del tipo empleado en la tarjeta de ampliación de puertos de entrada salida.

Circuito de tratamiento digital

Reloj

El reloj de punto CKP es generado con un multivibrador monoestable contenido en el circuito integrado U12. Cuando el biestable B4 (figura 1) lo permite (PERMISO = 1), la propia salida del monoestable excita su entrada, comportándose como un multivibrador estable. El

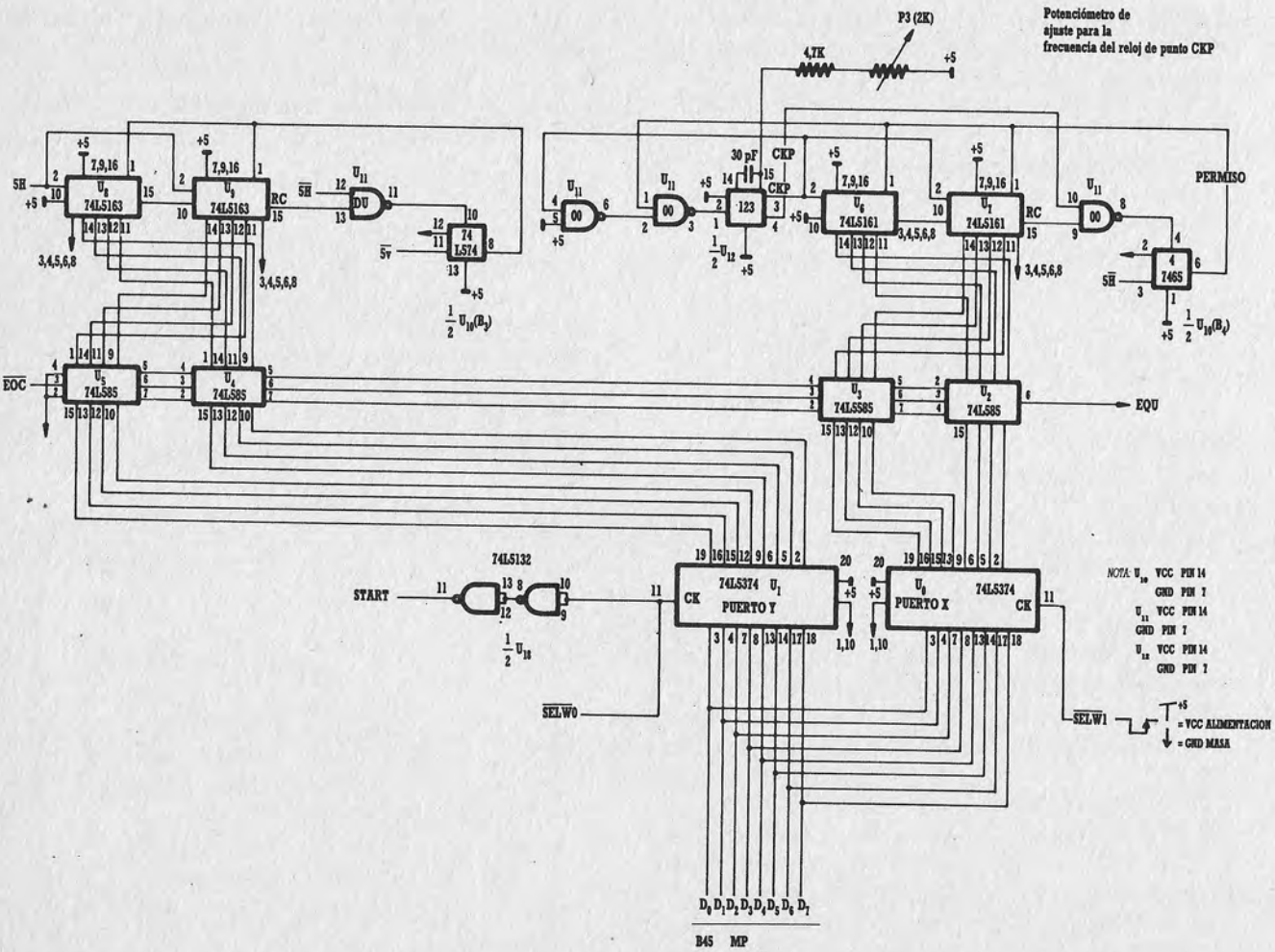


Fig. 1. Circuito digital: Reloj de punto, contadores, comparadores y puertos.

potenciómetro P3 se deberá ajustar para que la frecuencia de la señal generada sea de 5 MHz.

Contadores de posición

Los circuitos U8 y U9 son dos contadores de 4 bits encadenados que conforman el contador Y. Al llegar a la cuenta 255, la salida RC de U9 hace que el biestable B3 cambie de estado, reseteando (cuenta 0) el contador con el próximo pulso de SH y manteniéndolo en dicho estado hasta la recepción de un nuevo flanco de bajada de la señal de sincronismo vertical que hace que la salida del biestable B3 pase a valor «1» y permita de nuevo el contar los pulsos SH.

Los circuitos U6 y U7 conforman el contador X. Estos contadores se incrementan con los flancos positivos de la señal CKP de 5 MHz de frecuencia. A la llegada de pulso 255 de dicha señal, U7 genera un pulso en la salida RC, que pone a «0» la salida del biestable B4 que a su vez hace que las salidas del contador pasen al valor «0» y permanezcan en dicho estado hasta la llegada del próximo flanco negativo de la señal SH. Es importante observar la analogía existente entre las señales CKP y SH de este circuito y las SH y SV del anterior.

Comparador

Los circuitos integrados U2, U3, U4, U5 constituyen un comparador digital de palabras de 16 bits. Las salidas de los contadores de posición X e Y están conectadas a la sección B del comparador, estando la sección de entrada A conexionada con los 2 puertos de 8 bits X e Y.

Cuando ambas palabras son iguales y la comparación está permitida (EDC = «1»), la salida EQU de U2 se pone a «1». Esta señal se utiliza en el circuito de control como posteriormente se describirá.

Puertos de entrada y salida

Los puertos de salida X e Y (U1 y U0 en figura 1) se han realizado con los circuitos integrados 74LS374, constituido por 8 biestables síncronos de tipo D. Para la escritura en los mismos se ha conectado las señales -SELW0 y -SELW1, que son activas a nivel bajo a las entradas de reloj de los chips. El circuito decodificador de la figura 5 es el encargado de generar las señales de escritura anteriores a partir del bus de direcciones y las señales de lectura y escritura en puerto procedentes del microprocesador.

Como puerto de entrada para el microprocesador (U16 en figura 2), se utiliza también el circuito integrado 74LS374. En este registro se almacenan los 8 bits correspondientes a una digitalización. Para proceder a su lectura se ha de poner su entrada «OUTPUT CONTROL» a «0», lo que se consigue con su conexión a la línea SELR2, activa baja y generada también por el circuito decodificador.

Como puerto de entrada para leer el estado de la señal EOC (U17 en la figura 2) del circuito de control se utiliza el C.I. 74LS244. Para leer dicha variable es necesario poner a estado «0» las entradas de Enable de dicho chip, lo que se consigue conectando a éstas la salida de SELR3 del circuito decodificador.

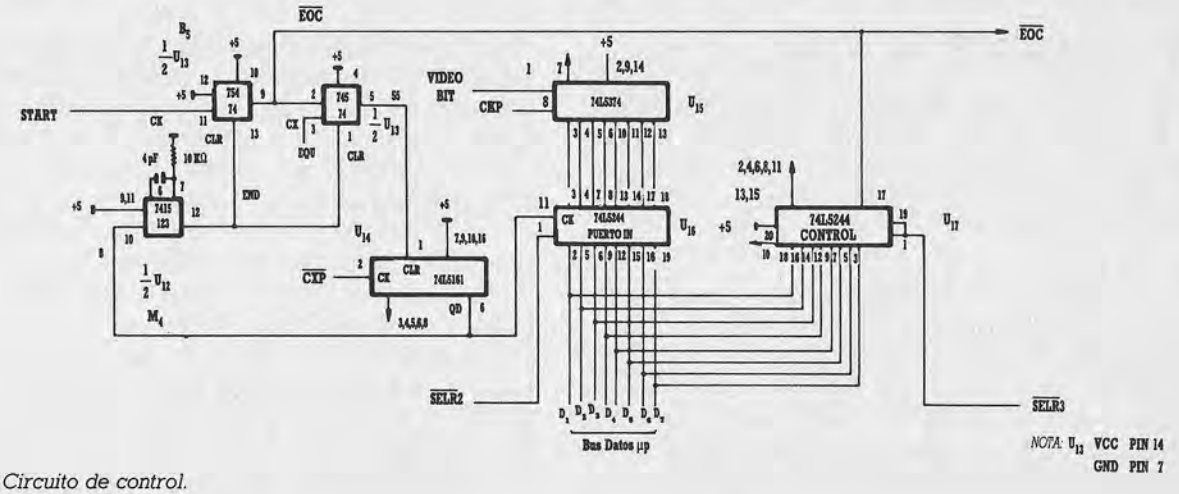


Fig. 2. Circuito de control.

Circuito de control

Los comentarios posteriores se harán con referencia a las figuras 2 y 3, donde se representan el circuito y el diagrama temporal de las señales a él asociadas.

La señal de escritura en el puerto Y se utiliza como señal de comienzo para la digitalización de los 8 pixels. El flanco positivo de este pulso pone a 1 el biestable B5, habilitando el comparador de 16 bits. Cuando los contadores de posición del pincel electrónico coinciden con las coordenadas escritas en los dos puertos de salida X e Y, el comparador genera un pulso (EQU) de una duración de 200 nanosegundos, que hace que el biestable B6 capture la salida del biestable 5 que anteriormente había puesto a 1 la señal START, haciendo que su salida SS se ponga a 1. Esta acción habilita el contaje de pulsos de reloj al contador de 4 bits U14. A la llegada del octavo pulso de reloj de salida Qd de dicho contador se pone a 1, haciendo que el registro de 8 bits U16 capture el dato procedente del registro o de desplazamiento U15 y que el monoestable M4 genere un pulso de 100 nanosegundos de duración. Este pulso pone a «0» la salida de los dos biestables B5 y B6, reseteando asimismo el contador, con lo que el circuito queda en condiciones de recibir otro pulso de START.

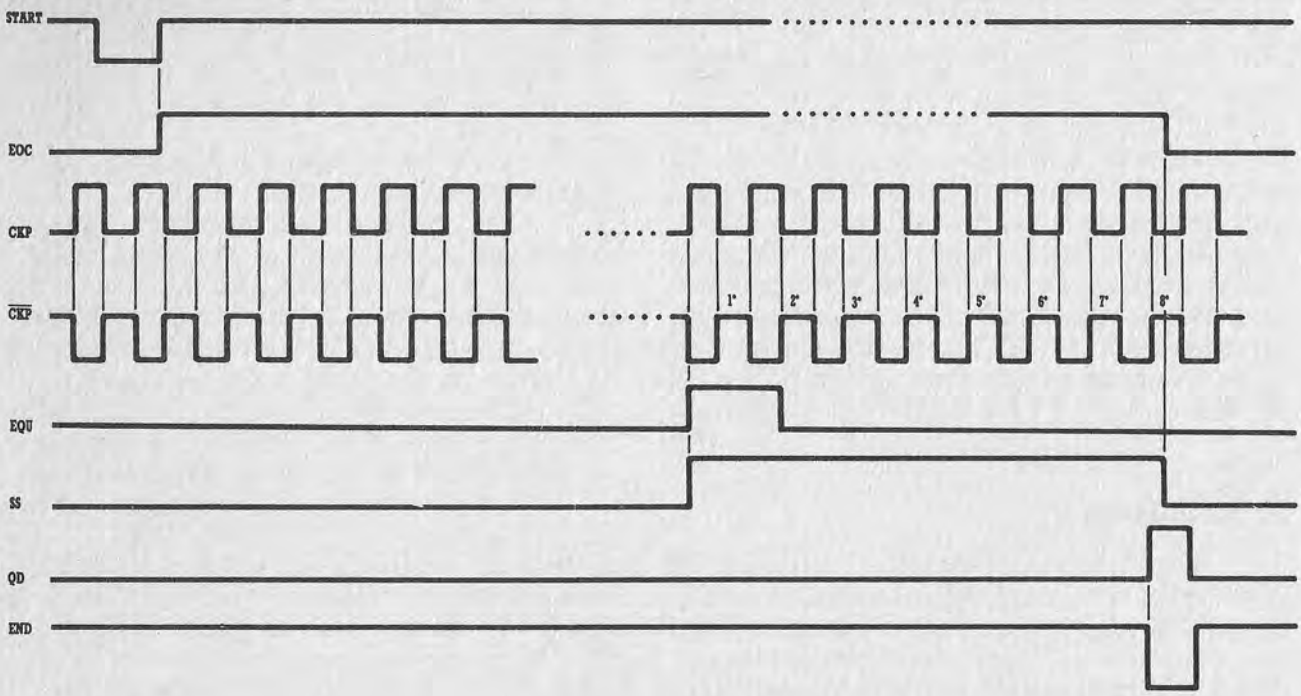


Fig. 3. Evolución de las señales en el circuito de control.

Programación de la tarjeta

El objetivo de este apartado es sentar las bases del programa que mediante la utilización de la circuitería hardware anteriormente definida digitalice la imagen y la presente en el monitor de alta resolución del ordenador personal.

Previamente a digitalizar la imagen deberá colocar el modo pantalla de su ordenador en modo de alta resolución y limpiar la pantalla. Para estas dos operaciones existen subrutinas en todos los sistemas operativos que pueden ser llamadas desde sentencias especiales desde lenguajes de alto nivel.

Es aconsejable que la mayoría de los programas de aplicación que utilicen esta tarjeta se escriban en lenguaje ensamblador. Esto es debido, por una parte, a la gran cantidad de información involucrada en una imagen (8 K octetos), y, por otro lado, a los condicionantes de tiempo existentes para una digitalización eficiente de la imagen. Las dos circunstancias anteriores hacen del uso de lenguajes de alto nivel poco eficaz. La rutina del ensamblador que deberá estar en todos los programas es la rutina para digitalizar ocho pixels. Esta rutina, cuyo programa se presenta a continuación, se ha escrito en ensamblador del microprocesador Intel 8088, aunque no

le será difícil traducirla al lenguaje ensamblador del microprocesador que contenga su ordenador personal.

Los nombres PUERTOX, PUERTOY, CONTROL y PUERTOIN del programa corresponden en la realidad con las direcciones físicas asignadas a los mismos mediante el circuito decodificador, como se explicó en un artículo anterior de esta misma serie:

```
          ; PROGRAMA DE LECTURA DE PIXEL

        MOV AL,COORDX     ; CARGA EN ACUMULADOR COORDENADA X
        OUT PUERTOX,AL    ; SACA POR PUERTO X COORDENADA X
        MOV AL,COORDY     ; CARGA EN ACUMULADOR COORDENADA Y
        OUT PUERTOY,AL    ; SACA POR PUERTOY COORDENADA Y
                          ; ARRANCA COMPARADOR

BUCLE   IN  AL,CONTROL    ; CARGA EN ACUMULADOR REGISTRO CONTROL
        AND AL,10000000B  ; ¿ TERMINADA DIGITALIZACION EOC=0 ?
        JNZ' BUCLE        ; NO ,SIGUE ESPERANDO
        IN AL,PUERTOIN    ; SI, CARGA EL VALOR OBTENIDO
```

La misión de esta rutina es fácil de comprender con las indicaciones anteriores. La rutina comienza por escribir, en primer lugar, la coordenada X en el puerto PUERTOX y, posteriormente, la coordenada Y en el puerto PUERTOY; la escritura de esta última coordenada habilita los comparadores y comenzará la búsqueda de las coordenadas escritas. La captura de los 8 pixels por parte de la tarjeta es indicada al efectuar una transición de «1» a «0» el bit más significativo del registro de entrada CONTROL, que está conectado, como anteriormente se indicó, a la señal EOC. Esta transición es detectada por las tres instrucciones que conforman el bucle de espera. Una vez se detecta que dicho bit está a «0», obtendremos en el acumulador AL un octeto cuyo bit más significativo corresponde al valor digital del pixel (X,Y). El bit menos significativo se corresponde con el valor del pixel de coordenadas (X+7,Y). El valor de coordenada X podrá estar comprendido entre 0 y 248 (téngase en cuenta que una coordenada X superior a 248 daría lugar a un funcionamiento incorrecto del circuito) y la coordenada Y entre 0 y 255.

Utilizando la rutina anterior se presenta un programa que digitaliza la imagen completa y la presenta en el monitor de un IBM-PC. El programa supone que los gráficos de alta resolución en blanco y negro (640*200) están activados. La zona de memoria gráfica en este caso comienza en el segmento X'800" para las filas pares y en X'BA00" para las impares. Estas direcciones tendrá que modificarlas según sea su ordenador. Además, esta memoria es bit-mapped, lo que quiere decir que cada octeto de la memoria se corresponde con 8 pixels consecutivos de la misma línea en pantalla, correspondiendo el más significativo al pixel izquierdo de la serie de ocho. Como se observa, el octecto obtenido en el puerto de entrada tiene la misma estructura que la de la memoria gráfica comentada.

Listado del programa

La misión del programa anterior es la de digitalizar una columna de 8 pixels de ancho por cada cuadro. Para ello utiliza la rutina de digitalización de 8 pixels 200 veces, con una «X» determinada y haciendo variar «Y» desde 32 hasta 232, con lo que se digitaliza únicamente la parte central de la imagen. Cada octeto obtenido es almacenado secuencialmente en un vector denominado TABLA y cuando se alcanza la ordenada 232 se traspasa dicho vector a la memoria gráfica en el tiempo que transcurre hasta que al pincel electrónico llegue de nuevo la ordenada 32, no perdiendo ningún cuadro. La transferencia se realiza teniendo en cuenta que los octetos que ocupan una posición par en el vector TABLA se almacenarán en el buffer de memoria gráfica correspondiente a las filas pares, haciendo la operación equivalente con los octetos que ocupan posiciones impares en el vector. El proceso anterior se repite desde X = 0 hasta X = 248 con incrementos de 8, finalizando el proceso en la última abscisa.

Cuando se escriba dicha rutina en el ensamblador del microprocesador que se posea, téngase en cuenta que si se pretende digitalizar la misma columna de 8 pixels en cada cuadro, el tiempo de que dispone para escribir coordenadas, obtener el dato y almacenarlo en memoria, así como la actualización de punteros y variables ha de ser menor que el tiempo de duración de una línea: 64 microsegundos. Si no es así, el programa tendrá que esperar al próximo cuadro, puesto que la traza habrá superado las coordenadas deseadas. Esto da lugar a que deberá escribir un programa optimizado para que su duración sea menor de 64 microsegundos en el caso de no poder conseguirlo, se podría digitalizar en un campo 8 pixels de cada fila par o impar, con lo que dispondría para realizar las operaciones anteriores de 128 microsegundos.

Como para los lenguajes de alto nivel es difícil calcular los tiempos de ejecución

porque se desconoce el código objeto generado, no es aconsejable su uso si se va a utilizar el hardware diseñado con su plena potencia.

Listado de salida del ensamblaje de un programa de lectura

```
1          ; Programa que digitaliza la señal de vídeo por columnas y
2          ; la presenta en memoria gráfica
3          ;-------------------------------
4          ; BX  -- puntero a vector TABLA y puntero a filas impares en vídeo-buffer
5          ; SI  -- puntero a TABLA
6          ; DI  -- puntero a filas pares vídeo-buffer
7          ; BP  -- coordenada x
8          ; CH  -- coordenada y
9          ;-------------------------------
10         ; Direcciones de los puertos de entrada y salida
11 = 0300          puertox  equ 300h ; puerto que contiene coordenada X
12 = 0301          puertoy  equ 301h ;puerto que contiene coordenada Y
13 = 0302          puertoin equ 302h ;puerto que contiene valor pixels
14 = 0303          control  equ 303h ;puerto que contiene señal EOC
15         ;--------------------------------
16         ; Segmento de datos
17 0000                                    dataseg                    segment
18 0000        C8 [                tabla        db      200   dup (?)
19                          ??
20                                    ]
21
22 00C8                                    dataseg    ends
23         ;--------------------------------
24         ; Segmento de memoria gráfica
25 0000                    vídeo          segment at 0b800h
26 0000                    buu     label  byte
27 0000                    vídeo          ends
28         ;--------------------------------
29         ; Segmento de código
30 0000                    codeseg                                    segment
31 0000                    main                proc                   far
32                         assume cs:codeseg,ds:datseg,es:Vídeo
33 0000            comienzo
34         ; Introduce en stack segmento y offset de datos
35 0000  1E            push ds
36 0001  B8 0000       mov ax,0
37 0004  50            push ax
38         ; Pone en Ds segmento de datos
39 0005  B8 ---- R     mov ax,dataseg
40 0008  8E D8         mov ds,ax
41         ; Pone en ES segmento vídeo-buffer
42 000A  B8 ---- R     mov ax,vídeo
43 000D  8E C0         mov es,ax
44         ;--------------------------------
45         ; Inicializaciones
46 000F  FC            cld        ; SI se incrementará
47 0010  BF 0000       mov di,0 ; DI puntero a vídeo-buffer
48 0013  BD 0000       mov bp,0 ; x=0 ,inicializa coordenada x
```

```
0016  B5 20                orden: mov ch,32 ; y=32, inicializa coordenada y
0018  8D 1E 0000 R         lea bx,tabla ; carga a dirección inicial tabla en BX
      ; Saca coordenadas por los puertos
001C  BA 0300              mov dx,puertox ;saca coordenada x por puerto A
001F  8B C5                mov ax,bp
0021  EE                   out dx,al
0022  BA 0301 saca:        mov dx,puerto y; saca coordenada y por puerto B
0025  8A C5                mov al,ch
0027  EE                   out dx,al                ; desinhibe tarjeta
      ; Espera a que se encuentre el punto EQU=1 y se digitalice EOC=1
0028  BA 0303              mov dx,control
002B  EC            espera: in al,dx
002C  24 80                and al,10000000b ; EOC en bit 7 de reg. control A
002E  75 FB                jnz espera
      ; Consigue vídeo-byte
0030  BA 0302              mov dx,puertoin
0033  EC                   in al,dx
      ; Almacénalo en tabla
0034  88 07                mov ds:[bx],al
0036  43                   inc bx                   ; bx apunta a próximo byte en TABLA
      ; Control de filas
0037  FE C5                inc ch                   ; próxima fila
0039  80 FD E8             cmp ch,232               ; realizadas las 200 filas ?
003C  75 E4                jnz saca                 ; no, vuelve con la ordenada siguiente
      ; Almacenamiento en vídeo-buffer
003E  57                   push di                  ; almacena puntero vídeo-buffer
003F  8D 36 0000 R almac: lea si,tabla ;carga con dirección inicial tabla
0043  8B DF                mov bx,di
0045  81 C3 2000           add bx,2000h             ; bx apunta a filas impares
0049  B9 0064              mov cx,100               ; 200 filas en video-buffer
004C  AD                allí: lodsw                 ; al -fila par, ah - fila impar
004D  26: 88 05            mov es:[di],al           ; almacénalo en fila par
0050  26: 88 07            mov es:[bx],al           ; almacénalo en fila impar
0053  83 C3 50             add bx,80                ; próxima fila impar
0056  83 C7 50             add di,80                ; próxima fila par
0059  E2 F1                loop allí
      ; Control abscisas
005B  5F                   pop di                   ; recupera puntero vídeo-buffer
005C  47                   inc di                   ; próxima columna en vídeo-buffer
      ; Control abscisas
005D  83 C5 08             add bp,8
0060  81 FD 0100           cmp bp,256
0064  75 B0                jnz orden
      ; Vuelve al DOS
0066  CB                      ret
0067                       main                     endp
0067                       codeseg                  ends
                           end    comienzo
```

Segments and groups:

Name	Size	align	combine class
CODESEG	0067	PARA	NONE
DATASEG	00C8	PARA	NONE
VIDEO	0000	AT	B800

Symbols:

Name	Type	Value	Attr
ALLI	L NEAR	004C	CODESEG
ALMAC	L NEAR	003F	CODESEG
BUU	L BYTE	0000	VIDEO
COMIENZO	L NEAR	0000	CODESEG
CONTROL	Number	0303	
ESPERA	L NEAR	002B	CODESEG
MAIN	F PROC	0000	CODESEG Length=0067
ORDEN	L NEAR	0016	CODESEG
PUERTOIN	Number	0302	
PUERTOX	Number	0300	
PUERTOY	Number	0301	
SACA	L NEAR	0022	CODESEG
TABLA	L BYTE	0000	DATASEG Length=00C8

Warning Errors	Severe Errors
0	0

Aplicaciones

La digitalización de imágenes a partir de señal de vídeo es un medio muy atractivo para diseñar paquetes de programas que presenten figuras del mundo real. La resolución de las pantallas de los ordenadores personales económicos es, en general, muy limitada, pero resulta suficiente para poder mantener en un solo medio mecanizado tanta información codificada como la de tipo gráfico e imágenes.

Para poder utilizar la información digitalizada de tipo binario es conveniente realizar un tratamiento previo, con objeto de reducir en lo posible el ruido causado en la digitalización por las imperfecciones del sistema. Un proceso elemental de imágenes es posible en el ordenador personal, que permita la extracción de la información contenida en las partes más importantes. Mediante promediado por zonas, diferenciación entre imágenes y cambios de contraste pueden generarse pantallas que presenten de forma más útil que con letras la información de interés para el usuario.

Estudio de las energías: energía potencial y cinética

UANDO se aplica una fuerza para mover un cuerpo de un punto a otro, a lo largo de un espacio, se realiza un trabajo. Para medirlo se utiliza la fórmula:

$T = F \cdot e$ (T = Newtons · metro = N · m = JULIOS)

1. *Energía y energía mecánica*

Al hablar de si un sistema es capaz o no de realizar un trabajo, estamos indicando con ello si ese sistema tiene o no energía. Considerando este punto de vista, la energía puede definirse como la capacidad que tiene un sistema para producir trabajo.

Sólo hay energía mecánica, que, a su vez, se divide en *energía potencial* y en *energía cinética.* Con frecuencia se habla de energía térmica, energía eléctrica, energía química..., pero todas estas energías no son más que formas de la energía mecánica, ya que al analizarlas en su aspecto microfísico, es decir, preocupándonos del mundo de las partículas que componen los sistemas, todas estas energías provienen de las energías mecánicas de dichas partículas.

Toda energía, por su trabajo acumulado, se mide en Julios (J).

2. *Energía potencial*

Todo sistema situado a una cierta altura tiene capacidad para realizar trabajo; así ocurre con el agua almacenada en un pantano. A este tipo de energía, que tiene almacenado un sistema debido a que se encuentra a una cierta altura, se le llama energía potencial.

La energía potencial de un sistema se libera cuando se le deja a éste que descienda toda o parte de su altura.

Un sistema no tiene una energía potencial determinada, sino que ésta depende del plano de referencia que se considere. Así un globo aerostático tiene diferente energía con respecto a la cumbre de una montaña que con respecto al nivel del mar.

Para calcular la energía potencial Ep que tiene un cuerpo de masa *m,* situada a una altura *h,* hay que calcular el trabajo que nos costará llevarlo a dicha altura. La fuerza que habrá que hacer será igual al peso, $P = m \cdot g$, y el espacio recorrido será la altura *h.* Por consiguiente:

$$T = Ep = F \cdot e = m \cdot g \cdot h \quad (g = 9{,}8\ m/s^2)$$

3. *Energía cinética*

Todo cuerpo o sistema, que tiene una velocidad, tiene almacenada una energía y, por tanto, es capaz de realizar un trabajo. Un balón inmóvil no es capaz de romper un cristal, pero si se le lanza a una elevada velocidad contra él, sí lo rompería.

Para calcular la energía cinética almacenada en un sistema de masa *m* a causa de su velocidad *v,* vamos a dar al sistema un trabajo, que quedará en él en forma de energía cinética. Al aplicarle una fuerza F el sistema se acelera, recorriendo un espacio $e = 1/2 \cdot a \cdot t^2$. Por consiguiente, el trabajo dado será:

$$T = F \cdot e = m \cdot a \cdot (1/2 \cdot a \cdot t^2) = 1/2 \cdot m \cdot (a \cdot t)^2 = 1/2 \cdot m \cdot v^2$$

En el movimiento producido por F se supone que no hay espacio inicial *e',* ni velocidad inicial *v',* con lo que las ecuaciones del movimiento uniformemente acelerado quedan reducidas así:

$$e = 1/2 \cdot a \cdot t^2; \quad v = a \cdot t$$

Según la expresión obtenida, la *energía cinética* tiene como fórmula:

$$Ec = 1/2 \cdot m \cdot v^2$$

```
10 REM *************************
12 REM *                       *
15 REM * ESTUDIO DE LA ENERGIA *
20 REM *                       *
25 REM *   POTENCIAL Y DE LA   *
30 REM *                       *
35 REM *   ENERGIA CINETICA.   *
40 REM *                       *
45 REM *************************
50 CLS
55 PRINT: PRINT
60 PRINT TAB(14);"ENERGIAS"
65 PRINT TAB(14);"---------"
70 PRINT :PRINT
80 PRINT TAB(3);"PUEDE CONSIDERARSE QUE LA ENER"
90 PRINT "GIA ES LA CAPACIDAD QUE TIENE UN"
100 PRINT "SISTEMA PARA PRODUCIR TRABAJO."
102 FOR I=1 TO 900: NEXT I
105 PRINT
110 PRINT TAB(3);"SOLO HAY ENERGIA MECANICA QUE"
120 PRINT "A SU VEZ SE DIVIDE EN:"
130 PRINT:PRINT TAB(3);"1.- ENERGIA POTENCIAL."
140 PRINT :PRINT TAB(3);"2.- ENERGIA CINETICA."
145 FOR I=1 TO 950: NEXT I
150 PRINT :PRINT:PRINT :PRINT
160 PRINT TAB(3);"PARA CONOCER CADA UNA DE ESTAS"
170 PRINT "ENERGIAS:"
175 PRINT
180 INPUT "PULSA LA OPCION DESEADA (1-2)";A
190 ON A GOTO 300,800
200 REM *********************
202 REM * EDICION DE FIN DE *
204 REM *  P R O G R A M A  *
206 REM *********************
210 CLS
220 PRINT :PRINT :PRINT :PRINT :PRINT
230 PRINT "DESEAS CONOCER OTROS VALORES DE"
240 PRINT "ENERGIAS (S/N):";
250 INPUT K$
260 IF K$="S" OR K$="s" THEN GOTO 50
270 IF K$<>"N" AND K$<>"n" THEN GOTO 210
280 CLS: PRINT :PRINT :PRINT :PRINT
285 PRINT :PRINT TAB(10);"¡¡¡ ADIOS !!!"
290 FOR I=1 TO 2000: NEXT I:CLS
295 END
300 REM *************************
302 REM * ESTUDIO DE LA ENERGIA *
304 REM *   P O T E N C I A L   *
306 REM *************************
310 CLS
320 PRINT :PRINT TAB(9);"ENERGIA POTENCIAL"
330 PRINT TAB(9);"------------------"
340 PRINT
350 PRINT TAB(3);"ES LA ENERGIA QUE TIENE AL"
360 PRINT "MACENADO UN SISTEMA DEBIDO A "
370 PRINT "QUE SE ENCUENTRA A UNA CIERTA"
380 PRINT "ALTURA <h>."
385 FOR I=1 TO 1500: NEXT I
390 PRINT :PRINT TAB(3);"SE LIBERA CUANDO SE DEJA A"
400 PRINT "ESTE QUE DESCIENDA TODA O PAR"
410 PRINT "TE DE SU ALTURA <h>."
420 FOR I=1 TO 1500: NEXT I
425 PRINT
430 PRINT TAB(3):"PARA CALCULAR SU VALOR ES NE"
440 PRINT "CESARIO DETERMINAR EL TRABAJO"
450 PRINT "QUE NOS CUSTARA LLEVAR EL CUER"
460 PRINT "PO DE MASA <m> A DICHA ALTURA."
465 FOR I=1 TO 1500:NEXT I
470 PRINT:PRINT "POR TANTO:"
480 PRINT :PRINT TAB(5);"Ep = m.g.h      (JULIOS)"
482 FOR I=1 TO 1500: NEXT I
485 PRINT
490 PRINT :INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
500 CLS
520 PRINT TAB(3);"DETERMINACION DEL VALOR DE"
530 PRINT "LA ENERGIA POTENCIAL:"
535 FOR I=1 TO 900: NEXT I
540 PRINT:PRINT
545 LET G=9.8
550 PRINT "MASA <m> DEL CUERPO,"
552 PRINT TAB(3);"EN kg:";
554 INPUT M
```

```
555 PRINT
560 PRINT "ALTURA <h> A LA QUE SE"
570 PRINT "ENCUENTRA DICHO CUERPO,"
575 PRINT TAB(3);"EN METROS (m):";
580 INPUT H
582 PRINT
585 PRINT "ACELERACION DE LA GRAVE"
588 PRINT "DAD: g = 9.8 m/s².
590 PRINT :PRINT :PRINT
595 FOR I=1 TO 1000: NEXT I
600 PRINT TAB(3);"LA ENERGIA POTENCIAL DE"
610 PRINT "DICHO CUERPO ES:"
620 LET E=M*G*H
630 PRINT :PRINT TAB(6);"Ep =";E;"JULIOS."
640 PRINT :PRINT
645 FOR I=1 TO 2000: NEXT I
650 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
660 GOTO 210
800 REM ************************
802 REM * ESTUDIO DE LA ENERGIA *
804 REM *    C I N E T I C A    *
806 REM ************************
810 CLS
820 PRINT :PRINT TAB(9);"ENERGIA CINETICA"
830 PRINT TAB(9);"----------------"
835 PRINT
840 PRINT TAB(3);"ES LA ENERGIA QUE TIENE AL"
850 PRINT "MACENADO UN SISTEMA DEBIDO A"
860 PRINT "QUE LLEVA UNA DETERMINADA VELO'
870 PRINT "CIDAD <v>."
880 FOR I=1 TO 1500: NEXT I
890 PRINT :PRINT  TAB(3);"PARA CALCULAR LA ENERGIA CI"
900 PRINT "NETICA, Ec, ALMACENADA EN UN "
910 PRINT "SISTEMA DE MASA <m> A CAUSA DE"
920 PRINT "SU VELOCIDAD, <v>, HAY QUE CAL"
930 PRINT "CULAR EL TRABAJO QUE HEMOS DADO"
940 PRINT "AL SISTEMA PARA QUE ALCANCE DI"
950 PRINT "CHA VELOCIDAD."
955 FOR I=1 TO 2500: NEXT I
960 PRINT :PRINT TAB(3);"AL APLICARLE UNA FUERZA, F,"
970 PRINT "EL SISTEMA SE ACELERA, CON LO"
980 PRINT "QUE RECORRE UN ESPACIO, e."
982 PRINT :PRINT :PRINT
984 FOR I=1 TO 3000:NEXT I
986 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
988 CLS
990 PRINT :PRINT :PRINT TAB(3);"POR TANTO, EL TRABAJO DADO"
1000 PRINT "DADO ES:
1010 PRINT :PRINT "T = F.e = 1/2.m.v²"
1015 FOR I=1 TO 1700: NEXT I
1020 PRINT :PRINT :PRINT TAB(3);"LA ENERGIA CINETICA TIENE"
1030 PRINT "POR FORMULA ENTONCES:"
1040 PRINT :PRINT "Ec = 1/2.m.v²  (JULIOS)"
1050 FOR I=1 TO 1500: NEXT I
1060 PRINT :PRINT
1070 PRINT "DONDE TENEMOS QUE:"
1080 PRINT :PRINT TAB(3);"Ec: ENERGIA CINETICA"
1090 PRINT TAB(3);" m: MASA DEL CUERPO EN Kg."
1100 PRINT TAB(3);"v²: VELOCIDAD DEL CUERPO."
1110 PRINT :PRINT:PRINT :PRINT
1115 FOR I=1 TO 2000: NEXT I
1120 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
1130 CLS
1140 PRINT:PRINT TAB(3);"DETERMINACION DEL VALOR DE"
1150 PRINT "LA ENERGIA CINETICA:"
1160 FOR I=1 TO 900: NEXT I
1170 PRINT :PRINT
1180 PRINT "MASA <m> DEL CUERPO,"
1190 PRINT TAB(3);"EN Kg:";
1200 INPUT M
1210 PRINT:PRINT
1220 PRINT "VELOCIDAD <v> QUE LLEVA"
1230 PRINT "DICHO CUERPO, EN m/s²:";
1240 INPUT V
1250 PRINT:PRINT :PRINT :PRINT
1270 PRINT TAB(3);"LA ENERGIA CINETICA DE"
1280 PRINT "DICHO CUERPO ES:"
1290 LET C=1/2*M*(V*V)
1300 PRINT :PRINT TAB(6);"Ep =";C;"JULIOS."
1310 PRINT :PRINT:PRINT
1320 FOR I=1 TO 2000: NEXT I
1330 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
1340 GOTO 210
```

Modificaciones para otros equipos

Tienes que realizar las modificaciones siguientes según tu ordenador:

ZX-SPECTRUM

```
190 IF A = 1 THEN GOTO 300 IF A = A
THEN GOTO 300
194 IF A = 2 THEN GOTO 800 IF A = A
THEN GOTO 800
196 CLS: GOTO 105
295 GOTO 9999
```

COMMODORE

```
50 PRINT CHR$ (147)
210 PRINT CHR$ (147)
280 PRINT CHR$ (147)
290 FOR I = 1 TO 2000: NEXT I: PRINT
CHR$ (147)
310 PRINT CHR$ (147)
500 PRINT CHR$ (147)
810 PRINT CHR$ (147)
988 PRINT CHR$ (147)
1130 PRINT CHR$ (147)
```

MATEMATICAS

Distancias en el espacio afín euclídeo (y II)

Como recordarás, con anterioridad ya se ha estudiado, en otro capítulo, el concepto de *distancia* en el espacio afín euclídeo (espacio afín real cuyo espacio vectorial es euclídeo). El estudio se centró en determinar el valor de las distancias para tres casos:

— Distancia existente entre dos puntos, <P> y <Q>.

— Distancia entre un punto, <P>, y una recta, <r>.

— Distancia entre dos rectas, <r> y <s>.

En este apartado continuaremos con distancias, pero introduciendo conceptos nuevos. Ante todo, lo primero es definir el concepto de PLANO en el espacio afín euclídeo:

$$\pi = \{x \in E(3) \,/\, x = P\ (\alpha \cdot \bar{u} + \beta \cdot \bar{v}),\ \text{con P fijo}$$

$$(\bar{u}, \bar{v})\ \text{independientes, y}\ \alpha\ \beta\ \Sigma\ R)$$

Consecuentemente, es posible llegar a calcular nuevos valores para las distancias:

— Distancia de un punto, P, a un plano, <π>.

— Distancia de una recta, <r>, a un plano <π>.

— Distancia entre dos planos, <π> y <δ>.

1. *Distancia de un punto a un plano*

La distancia de un punto a un plano $\pi \equiv Ax + By + Cz + D = 0$, viene dada, geométricamente, por la perpendicular desde el punto P exterior que corte a un punto del plano <π>; la distancia entre ambos puntos es la que buscamos.

Se determina calculando el producto escalar del vector $\overrightarrow{PP}$, y el vector unitario *n'* que tenga la misma dirección que el vector asociado al plano *n* (A, B, C).

El vector unitario *n'* se obtiene dividiendo las componentes del vector asociado por el módulo del mismo:

$$n' = \left| \frac{A}{|n|}, \frac{B}{|n|}, \frac{C}{|n|} \right|$$

De donde la distancia del punto P (X1, Y1, Z1) al plano <π> es:

$$d\,(P', \pi) = PP' \cdot n' = |PP'| \cdot |n'| \cdot \cos\alpha = |PP'| \cdot \cos\alpha \ \text{(ya que } |n'| = 1)$$

Expresando el producto escalar en función de las componentes de los vectores obtenemos:

$$d\,P', \sigma) = (X1 - X, Y1 - Y, Z1 - Z) \cdot \left(\frac{A}{|n|}, \frac{B}{|n|}, \frac{C}{|n|} \right) =$$

$$= (X1 - X) \cdot \frac{A}{|n|} + (Y1 - Y) \cdot \frac{B}{|n|} + (Z1 - Z) \cdot \frac{C}{|n|} =$$

$$= \frac{A \cdot (X1 - X) + B \cdot (Y1 - Y) + C \cdot (Z1 - Z)}{\sqrt{(A^2 + B^2 + C^2)}} =$$

$$= \frac{A \cdot X1 + B \cdot Y1 + C \cdot Z1 - A \cdot X - B \cdot Y - C \cdot Z}{\sqrt{(A^2 + B^2 + C^2)}} = \qquad [1]$$

Como P es un punto del plano <π), satisface la ecuación, con lo cual:

$$Ax + By + Cz + D = 0 \rightarrow D = Ax + By + Cz$$

sustituyendo en [1] resulta:

$$d\,(P', \pi) = \frac{AX1 + BY1 + CZ1 + D}{\sqrt{(A^2 + B^2 + C^2}} \qquad [2]$$

Cuando la distancia que se busca es la del origen de coordenadas O (0, 0, 0) al plano <π>, la fórmula [2] queda:

$$d\,(0, \pi) = \frac{A \cdot 0 + B \cdot 0 + C \cdot 0 + D}{\sqrt{(A^2 + B^2 + C^2)}} = \frac{D}{\sqrt{(A^2 + B^2 + C^2)}}$$

2. *Distancia entre recta y plano*

La distancia entre recta y plano se obtiene calculando la distancia entre la recta

<r> y un punto P que pertenezca al plano. Su valor viene dado por el módulo del producto vectorial del vector PP (X1 - X, Y1 - X, Z1 - Z) y el vector unitario v que tenga la misma dirección que el vector director de la recta, v (a, b, c).

$$v' = \left(\frac{a}{|v|}, \frac{b}{|v|}, \frac{c}{|v|}\right)$$

Y la distancia viene determinada por:

$|d(PP', v') = |PP'| \cdot |v'| \cdot \text{sen}\,\alpha = |PP' \cdot \text{sen}\,\alpha$ (ya que $|v'| = 1$)

Expresada en función de las componentes de los vectores:

$$d = \text{mod}\begin{vmatrix} i & j & k \\ X1-X & Y1-Y & Z1-Z \\ a & b & c \\ \hline \sqrt{(a^2+b^2+c^2)} & \sqrt{(a^2+b^2+c^2)} & \sqrt{(a^2+b^2+c^2)} \end{vmatrix}$$

3. *Distancia entre planos*

Sería la distancia de un punto cualquiera de uno de ellos al otro plano. Se reduce entonces al primer caso (distancia de un punto a un plano).

```
10 REM ****************************
15 REM *                          *
20 REM * DISTANCIAS EN EL ESPACIO *
25 REM *                          *
30 REM *      AFIN EUCLIDEO       *
35 REM *                          *
40 REM ****************************
45 CLS
50 PRINT TAB(7);"DISTANCIAS EN EL ESPACIO"
60 PRINT TAB(7);"----------------------------"
70 FOR I=1 TO 500: NEXT I
75 PRINT
80 PRINT "ESTE PROGRAMA TE PERMITE DETERMI-"
90 PRINT "NAR EL VALOR DE LA DISTANCIA EN"
100 PRINT "TRE CUALQUIERA DE LOS ELEMENTOS"
110 PRINT "DEL ESPACIO AFIN PARTIENDO DEL"
111 PRINT "PREVIO CONCEPTO DE PLANO."
112 FOR I=1 TO 1100: NEXT I
115 PRINT:PRINT
150 PRINT :PRINT "1. DISTANCIA DE UN PUNTO p A UN"
152 PRINT TAB(4);"PLANO <π>."
155 PRINT: PRINT "2. DISTANCIA ENTRE UNA RECTA r"
156 PRINT TAB(4);"Y UN PLANO <π>."
158 PRINT :PRINT "3. DISTANCIA ENTRE PLANOS."
166 PRINT :PRINT:PRINT :PRINT
170 FOR I=1 TO 2000: NEXT I
180 PRINT TAB(4);"PULSA LA OPCION DESEADA:"
190 PRINT TAB(6)"(1-2-3)";
200 INPUT Z
210 ON Z GOTO 900,500,1100
220 CLS
222 REM ****************************
224 REM * EDICION FIN DE PROGRAMA *
226 REM ****************************
230 PRINT :PRINT :PRINT :PRINT :PRINT
235 PRINT TAB(3);"DESEAS CONOCER EL VALOR DE"
240 PRINT TAB(3);"OTRA DISTANCIA (S/N):";
245 INPUT K$
250 IF K$="S" OR K$="s" THEN CLS: GOTO 50
255 IF K$<>"N" AND K$<>"n" THEN GOTO 220
260 CLS
265 PRINT :PRINT :PRINT :PRINT :PRINT
270 PRINT :PRINT TAB(10);"¡¡¡ ADIOS !!!"
275 FOR I=1 TO 2000: NEXT I
280 CLS
290 END
500 REM ************************
502 REM * DISTANCIA DE UN PUNTO *
503 REM * DEL PLANO <π> RESPEL- *
504 REM *   TO DE LA RECTA <r>  *
506 REM ************************
510 CLS
520 GOSUB 2000
530 GOSUB 2400
540 LET U1=V2*(Z1-Z0)-V3*(Y1-Y0)
550 LET U2=V3*(X1-X0)-V1*(Z1-Z0)
560 LET U3=V1*(Y1-Y0)-V2*(X1-X0)
570 LET P=SQR(U1*U1+U2*U2+U3*U3)
580 LET D=P/N1
590 CLS
```

```
600 PRINT :PRINT :PRINT :PRINT :PRINT
610 PRINT TAB(3);"LA DISTANCIA ENTRE EL PUNTO <p>"
615 PRINT "PERTENECIENTE AL PLANO <π> RES"
620 PRINT "PECTO DE LA RECTA <r> ES:"
630 PRINT :PRINT TAB(7);D;"UNIDADES."
635 PRINT :PRINT :PRINT :PRINT
640 FOR I=1 TO 2000:NEXT I
650 PRINT :PRINT :PRINT :PRINT
660 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
670 GOTO 220
900 REM ************************
902 REM * DISTANCIA DE UN PUNTO *
904 REM *     A UN PLANO <π>     *
906 REM ************************
910 CLS
920 GOSUB 2000
930 GOSUB 2800
940 LET P=A1*X0+B1*Y0+C1*Z0+D1
950 LET D=ABS(P/N1)
960 CLS
970 PRINT :PRINT :PRINT :PRINT :PRINT
980 PRINT TAB(3);"LA DISTANCIA DEL PUNTO <p> AL PLA"
990 PRINT "NO <π> ES:
1000 PRINT :PRINT TAB(7);D;"UNIDADES."
1010 PRINT :PRINT :PRINT :PRINT
1015 FOR I=1 TO 2000: NEXT I
1018 PRINT :PRINT :PRINT :PRINT
1020 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
1030 GOTO 220
1100 REM ***************************
1110 REM * DISTANCIA ENTRE PLANOS *
1120 REM ***************************
1130 CLS
1140 GOSUB 2800
1150 GOSUB 3000
1160 LET U1=B1*C2-C1*B2
1170 LET U2=C1*A2-A1*C2
1180 LET U3=A1*B2-B1*A2
1190 LET M=SQR(U1*U1+U2*U2+U3*U3)
1200 IF M=0 THEN GOTO 1300
1210 CLS
1220 PRINT :PRINT :PRINT :PRINT :PRINT
1230 PRINT TAB(3);"LA DISTANCIA ENTRE LOS DOS PLA"
1240 PRINT "NOS ES CERO, YA QUE SON SECANTES."
1250 PRINT :PRINT :PRINT :PRINT
1260 FOR I=1 TO 2000: NEXT I
1265 PRINT :PRINT :PRINT :PRINT
1270 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
1280 GOTO 220
1300 REM *** CASO DE PLANOS PARALELOS ***
1310 LET D=ABS((D1/N1)-(D2/N2))
1320 PRINT :PRINT :PRINT :PRINT :PRINT
1330 PRINT TAB(3);"LOS DOS PLANOS SON PARALELOS"
1340 PRINT "Y LA DISTANCIA ENTRE ELLOS ES:"
1350 PRINT :PRINT TAB(7);D;"UNIDADES."
1360 PRINT :PRINT :PRINT :PRINT
1370 FOR I=1 TO 2000: NEXT I
1375 PRINT :PRINT :PRINT :PRINT
1380 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
1390 GOTO 220
2000 REM ************************
2002 REM *    COORDENADAS DEL    *
2004 REM *      PUNTO, <p>.      *
2006 REM ************************
2010 CLS
2020 PRINT :PRINT :PRINT :PRINT :PRINT
2030 PRINT "COORDENADAS  DEL PUNTO p:"
2040 PRINT "p(X0,Y0,Z0)";
2050 INPUT X0,Y0,Z0
2060 RETURN
2400 REM ************************
2402 REM * COORDENADAS DEL VECTOR *
2404 REM *     DIRECTOR <v> DE    *
2406 REM *      LA RECTA, <r>.    *
2408 REM ************************
2410 CLS
2420 PRINT :PRINT :PRINT :PRINT
2430 PRINT "COORDENADAS DEL VECTOR DIRECTOR"
2440 PRINT "DE LA RECTA <r>."
2450 INPUT "v(V1,V2,V3)";V1,V2,V3
2460 LET N1=SQR(V1*V1+V2*V2+V3*V3)
2470 PRINT :PRINT :PRINT
2480 PRINT "COORDENADAS DE UN PUNTO PERTENE"
```

```
2490 PRINT "CIENTE A LA RECTA <r>."
2500 INPUT "PUNTO t:";X1,Y1,Z1
2510 RETURN
2800 REM ************************
2802 REM *  DATOS DEL PLANO <π>  *
2804 REM ************************
2810 CLS
2815 PRINT :PRINT :PRINT :PRINT
2820 PRINT TAB(3);"EL PLANO <π> TIENE DE ECUA
2830 PRINT "CION:"
2840 PRINT :PRINT TAB(3);"Ax + By + Cz + D = 0"
2845 FOR I=1 TO 800: NEXT I
2850 PRINT :PRINT "DONDE <A,B,C> SON LAS COOR"
2860 PRINT "DENADAS DE SU VECTOR NORMAL."
2865 FOR I=1 TO 800:NEXT I
2870 PRINT
2875 PRINT "COORDENADAS DEL VECTOR NORMAL"
2880 PRINT "m(A,B,C):";
2890 INPUT A1,B1,C1
2900 LET N1=SQR(A1*A1+B1*B1+C1*C1)
2910 RETURN
3000 REM ************************
3002 REM *  DATOS DEL PLANO <σ>  *
3004 REM ************************
3010 CLS
3015 PRINT :PRINT :PRINT :PRINT
3020 PRINT TAB(3);"EL PLANO <σ> TIENE DE ECUA"
3030 PRINT "CION:"
3040 PRINT :PRINT TAB(3);"A'x +B'y + C'z + D' = 0"
3045 FOR I=1 TO 800: NEXT I
3050 PRINT :PRINT "DONDE <A',B',C'> SON LAS"
3060 PRINT "COORDENADAS DE SU VECTOR NORMAL."
3065 FOR I=1 TO 800:NEXT I
3070 PRINT
3075 PRINT "COORDENADAS DEL VECTOR NORMAL"
3080 PRINT "m'(A',B',C')";
3090 INPUT A2,B2,C2
3100 LET N2=SQR(A2*A2+B2*B2+C2*C2)
3110 RETURN
```

Modificaciones para otros equipos

ZX-SPECTRUM

210 IF Z = 1 THEN GOTO 900 IFZ = 1 THEN GOTO 900
212 IF Z = 2 THEN GOTO 500 IFZ = 2 THEN GOTO 500
214 IF Z = 3 THEN GOTO 1100 IFZ = 3 THEN GOTO 110
216 CLS: GOTO 166
290 GOTO 9999

COMMODORE

45 PRINT CHR$ (147)
220 PRINT CHR$ (147)
260 PRINT CHR$ (147)
280 PRINT CHR$ (147)
510 PRINT CHR$ (147)
590 PRINT CHR$ (147)
910 PRINT CHR$ (147)
960 PRINT CHR$ (147)
1130 PRINT CHR$ (147)
1210 PRINT CHR$ (147)
2010 PRINT CHR$ (147)
2410 PRINT CHR$ (147)
2810 PRINT CHR$ (147)
3010 PRINT CHR$ (147)

El arte en la España musulmana

La manifestación más importante del arte hispanomusulmán es la arquitectura. En su evolución se distinguen tres períodos: el cordobés o califal (siglos VIII al X); el de los Reinos de Taifas, almorávide y almohade (siglo XI a mediados del XIII) y el granadino o nazarí (siglos XIII al XV).

Primer período (cordobés o califal). Entre sus elementos arquitectónicos destaca el *arco de herradura,* más peraltado que el visigodo, del cual procede. En estos arcos alternan dovelas blancas y rojas, según el procedimiento utilizado ya por los romanos en el acueducto de los Milagros de Mérida. Emplea otros tipos de arcos, como el de lóbulos, el de herradura apuntado y los arcos entrelazados. El arco va encuadrado por un alfiz, que deja en la parte superior unos triángulos esféricos denominados albanegas. La arquitectura califal utiliza todos los sistemas de bóvedas conocidos, siendo la más característica la llamada *bóveda califal,* formada por gruesos nervios que no se cruzan en su centro. La decoración es de *ataurique* (ornamentación de tipo vegetal) y geométrica, todavía muy simple a base de cuadrados y de rectángulos. Las dos obras

más importantes son la Mezquita de Córdoba y el Palacio de Medina Azahara.

— *Mezquita de Córdoba.* Construida entre los siglos VIII y XI, y en sus diversas ampliaciones intervinieron Abderramán I, II y III, Alhaquén II y Almanzor. Es de planta rectangular, con diecinueve naves separadas por columnas muy variadas, aprovechadas muchas de ellas de iglesias cristianas. Sobre la columna se eleva un pilar y ambos sostienen dos arcos superpuestos, el superior de medio punto y el inferior de herradura, ambos con alternancia de color en las dovelas. La parte más lujosa de la mezquita es el mihrab, realizado por Alhaquén II.

— *Palacio de Medina Azahara.* Situado a pocos kilómetros de Córdoba, fue mandado construir por Abderramán III para su favorita Zahara y completado por Alhaquén II.

Segundo período (Reinos de Taifas, almorávide y almohade). A partir de 1031, al desintegrarse el Califato de Córdoba, el arte continuó desarrollándose en los reinos de Taifas. El edificio más importante de este período es el palacio de la *Aljafería de Zaragoza,* construido por los Beni Hud en el siglo XI. Se caracteriza por su gran riqueza decorativa, el alargamiento del capital y el arco mixtilíneo. La invasión almohade supuso una reacción contra la lujosa decoración de los reinos de Taifas e introdujo una serie de novedades arquitectónicas, como la decoración a base de *listeles paralelos.* Al arte almohade pertenecen la Giralda de Sevilla, que era el alminar de la antigua mezquita, y la Torre del Oro.

Tercer período (granadino o nazarí). Granada fue la capital y residencia de los reyes nazaríes desde mayo de 1238 hasta enero de 1492. Los reyes nazaríes Mohamed I, Yusuf I y Mohamed V levantaron los tres palacios que constituyen el conjunto principal de la *Alhambra:* el Mexuar o Cuarto Dorado (Justicia); el Serrallo o cuarto de Comares (Palacio oficial); y el Harem o Cuarto de los Leones (Palacio privado). Aunque cada uno de estos palacios tienen en común un patio con fuentes o alberca en su centro, sin embargo, se diferencian entre sí notablemente, ya que mientras el de Comares es típicamente musulmán, el de los Leones tendría influencias cristianas, asemejándose a un claustro románico. La arquitectura civil nazarí es frágil y en ella predomina lo decorativo frente a lo constructivo. Los maravillosos arcos que adornan los patios y salones no sostienen en realidad nada. Los palacios granadinos más que edificios constituyen espléndidos conjuntos escenográficos. Se utilizan diversos tipos de arcos, siendo el más original el *arco de pabellón.* Tiene una profusa decoración a base de alicatados (especie de mosaico en que cada elemento decorativo es una pieza independiente de barro vidriado), atauriques, lacerías (decoración geométrica compuesta por líneas o cintas que se entrelazan formando polígonos o estrellas ordenados simétricamente) y ajaracas (ornamentación compuesta de ataurique y lacerías). Las numerosas columnas de sus patios ofrecen múltiples puntos de vista y magníficos juegos de perspectiva. Es extraordinaria la fusión del arte con la naturaleza por medio de fuentes y surtidores, albercas rodeadas de arrayanes y frondosos jardines.

— *El Generalife.* Fue una de las fincas de recreo de los reyes nazaríes que se encuentra situado muy próximo a la Alhambra. Constituye, en realidad, el mejor ejemplar de jardín árabe medieval que se conoce en la actualidad. El Generalife sorprende no tanto por sus construcciones, como por la belleza de los jardines y por los juegos de agua de las fuentes, como las del patio de la alberca. Era un palacio mirador: sencillo y con escasos aposentos.

```
10 REM ********************************
15 REM *   CONOCIMIENTOS SOBRE ARTE   *
20 REM *        HISPANO-MUSULMAN      *
25 REM ********************************
30 CLS
40 LET A=0
50 RANDOMIZE TIMER
60 DIM A$(10): DIM B$(40):DIM N(10)
70 REM ***************************
75 REM *  LECTURA DE PREGUNTAS   *
80 REM ***************************
90 FOR I=1 TO 10
100 READ A$(I)
110 NEXT I
120 REM ***************************
125 REM * LECTURA DE RESPUESTAS *
130 REM ***************************
140 FOR J=1 TO 40
```

```
150 READ B$(J)
160 NEXT J
170 REM ************************
175 REM * LECTURA DE SOLUCIONES *
180 REM *    A LAS RESPUESTAS    *
185 REM ************************
190 FOR K=1 TO 10
195 READ N(K)
200 NEXT K
210 REM ************************
215 REM * EDICION EN PANTALLA DE *
220 REM * PREGUNTAS Y RESPUESTAS *
225 REM ************************
230 FOR I=1 TO 10
240 PRINT:PRINT TAB(2);A$(I)
250 PRINT:PRINT
260 PRINT:PRINT TAB(6);B$(I*4-3)
270 PRINT:PRINT TAB(6);B$(I*4-2)
280 PRINT:PRINT TAB(6);B$(I*4-1)
290 PRINT:PRINT TAB(6);B$(I*4)
300 PRINT:PRINT
310 REM *********************
315 REM * INTRODUCCION DE LA *
320 REM * SOLUCION CORRECTA  *
325 REM *********************
330 INPUT "LA RESPUESTA ES: ";R
340 IF R=N(I) THEN GOSUB 500
350 IF R<>N(I) THEN GOSUB 800
360 NEXT I
370 PRINT:PRINT:PRINT
380 REM ************************
385 REM * EDICION DEL FIN DEL *
390 REM *      PROGRAMA        *
395 REM ************************
400 PRINT TAB(3);"RESPUESTAS CORRECTAS: ";A
410 PRINT:PRINT:PRINT
415 IF A<1 OR A>10 THEN
420 IF A>=1 AND A<4 THEN                    DEFICIENTE":GOTO 490
430 IF A=4 THEN PRINT "INSU                 GOTO 490
440 IF A=5 THEN PRINT "SUFICIE                  490
450 IF A=6 THEN PRINT "BIEN":GOTO
460 IF A>6 AND A<9 THEN PRINT "NOTABLE":GOTO 490
470 IF A=9 THEN PRINT "SOBRESALIENTE":GOTO 490
480 PRINT "MUY BIEN. PERFECTO. TIENES UN 10 DE NOTA"
490 FOR I=1 TO 2000:NEXT I:CLS:END
500 REM ************************
505 REM * SUBRUTINA DE RESPUESTAS *
510 REM *       CORRECTAS        *
515 REM ************************
520 CLS
530 PRINT:PRINT:PRINT
540 PRINT " ¡SOLUCION CORRECTA!"
550 LET A=A+1
560 FOR F=0 TO 2000:NEXT F:CLS
570 RETURN
800 REM ************************
805 REM * SUBRUTINA DE RESPUESTAS *
810 REM *      INCORRECTAS       *
815 REM ************************
820 CLS
830 PRINT:PRINT:PRINT
840 PRINT " ¡NO ES CORRECTO!"
850 PRINT:PRINT
860 PRINT "LA SOLUCION CORRECTA ES: "
865 PRINT:PRINT B$(I*4+N(I)-4)
870 FOR F=1 TO 5000: NEXT F:CLS
880 RETURN
1000 REM ************************
1010 REM * DATAS CON LAS PREGUNTAS *
1020 REM *   RESPUESTAS Y CON LAS  *
1030 REM *   SOLUCIONES CORRECTAS  *
1040 REM ************************
1050 DATA "PERIODOS DE LA ARQUITECTURA HISPANOMUSULMANA:","A QUE PERIODO PERTENE
CE LA GIRALDA DE SEVILLA","OBRA DEL PERIODO CORDOBES O CALIFAL"
1060 DATA "LA MEZQUITA DE CORDOBA ES DEL PERIODO...","QUE PERTENECE A LA ALHAMBR
A DE GRANADA:","QUIEN PARTICIPO EN LA AMPLIACION DE LA MEZQUITA DE CORDOBA"
1070 DATA "LA ALJAFERIA DE ZARAGOZA A QUE PERIODO PERTENECE:","COMO EJEMPLAR DE
JARDIN ARABE MEDIEVAL DESTACA:"
1080 DATA "LA DECORACION DE ALICATADOS, ATAURIQUES, LACERIAS Y AJARACAS ESTA PRE
SENTE SOBRE TODO EN:","QUE ELEMENTO ARQUITECTONICO DESTACA DEL PERIODO CORDOBES:
"
1090 DATA "1. CORDOBES, MOZARABE, MUDEJAR","2. CORDOBES, TAIFAS, NAZARI","3. TAI
```

```
FAS, NAZARI, MOZARABE","4. NAZARI, TAIFAS, MUDEJAR"
1100 DATA "1. CORDOBES","2. MUDEJAR","3. NAZAR","4. ALMOHADE","1. PALACIO DE MEI
INA AZZAHARA","2. GENERALIFE","3. TORRE DEL ORO","4. ALHAMBRA DE GRANADA"
1110 DATA "1. GRANADINO","2. TAIFAS","3. CORDOBES O CALIFAL","4. ALMOHADE","1. G
IRALDA DE SEVILLA","2. ALJAFERIA","3. CUARTO DE COMARES","4. GENERALIFE"
1120 DATA "1. ALHAQUEM II","2. BENI HUD","3. MOHAMED I","4. YUSUF I","1. NAZARI"
,"2. GRANADINO","3. CORDOBES O CALIFAL","4. REINOS DE TAIFAS"
1130 DATA "1. MEXUAR","2. GENERALIFE","3. TORRE DEL ORO","4. CUARTO DE LOS LEONE
S","1. GENERALIFE","2. MEZQUITA DE CORDOBA","3. ALHAMBRA DE GRANADA","4. GIRALDA
 DE SEVILLA"
1140 DATA "1. ALBANEGAS","2. ARCO DE HERRADURA","3. LACERIAS Y AJARACAS","4. ATA
URIQUES"
1150 DATA 2,4,1,3,3,1,4,2,3,2
```

Modificaciones para otros equipos

Tienes que desarrollar las siguientes modificaciones según el ordenador que tengas:

ZX-SPECTRUM

50 (NO SE PONE)

60 DIM A$ (10,40), DIM B$ (40,30), DIM N(10)

490 FOR I = 1 TO 1000: NEXT I: CLS: GOTO 9999

500 FOR F = 1 TO 1000: NEXT I: CLS

870 FOR F = 1 TO 3000: NEXT I: CLS

COMMODORE

30 PRINT CHR$ (147)

490 FOR I = 1 TO 2000: NEXT I: PRINT CHR$ (147): END

500 FOR F = 1 TO 2000: NEXT I: PRINT CHR$ (147)

870 FOR F = 1 TO 5000: NEXT I: PRINT CHR$ (147)

PARA LOS MAS PEQUEÑOS

División de polinomios: división entre (x - a). Regla de Ruffini

En el anillo de los polinomios en R [x] es posible definir la división de polinomios. Al operar podemos encontrarnos con divisiones exactas y otras que dan resto. Estas divisiones se realizan como una división normal donde el dividendo y el divisor son dos polinomios.

Cuando el polinomio divisor es del tipo (x - a), la división se puede hacer de manera muy cómoda, usando la *Regla de Ruffini,* que maneja sólo coeficientes de los polinomios dividendo y divisor:

Un ejemplo sería:

$$(x^4 - 3x^2 + 2) / (x - 3);$$

	1	0	-3	0	2	
3		3	9	18	54	
	1	3	6	18	56	= R

Regla de Ruffini

a) El grado del cociente es menor en una unidad que el del dividendo.

b) El primer coeficiente del cociente es el mismo que el primer coeficiente del dividendo.

c) Los otros coeficientes se obtienen multiplicando el anterior por <a>, sumando el coeficiente correspondiente del dividendo.

d) El resto es igual al producto del último coeficiente por <a>, más el término independiente del dividendo.

Teorema del resto. Teorema de Ruffini. Ceros de un polinomio

A) *Teorema del resto.* El resto de la división de un polinomio P (x) por (x - a) es igual al valor numérico del polinomio para x = a.

$$P(x) = (x - a) \cdot C(x) + R$$

B) *Teorema de Ruffini.* La condición necesaria y suficiente para que un polinomio P (x) sea divisible por (x - a) es que P (a) = 0.

$$C(x) = \frac{P(x)}{x - a} \Leftrightarrow P(a) = 0$$

C) *Ceros de un polinomio.* Se llaman ceros o raíces de un polinomio a los números reales <a> para los que su valor numérico es cero. Es decir:

$$a \in R \text{ es cero de } P(x) \Leftrightarrow P(a) = 0$$

Sobre el programa

El programa que se te presenta a continuación realiza la división de un polinomio P (x) entre el binomio (x - a), donde tú introduces el grado y coeficientes del polinomio por un lado y además introduces el valor del término <a> del binomio. Al final obtienes el polinomio cociente expresado por el método de Ruffini y donde se te indica si el resto es R = 0 o bien R <> 0.

```
10 REM ****************************
12 REM *                          *
15 REM * APLICACION DE LA REGLA DE *
20 REM *                          *
30 REM *  RUFFINI PARA DIVIDIR UN  *
35 REM *                          *
40 REM * POLINOMIO POR UN BINOMIO  *
45 REM *                          *
48 REM ****************************
50 CLS
55 PRINT :PRINT :PRINT
60 PRINT TAB(9);"REGLA DE RUFFINI"
70 PRINT TAB(9);"----------------"
80 PRINT :PRINT
90 PRINT TAB(3);"ESTE PROGRAMA TE PERMITE DETER"
100 PRINT "MINAR EL COCIENTE DE UN POLINOMIO"
110 PRINT "P(x) POR UN BINOMIO (x-a)."
120 PRINT :PRINT TAB(3);"SE OBTIENE EL POLINOMIO COCIEN"
130 PRINT "TE C(x) Y EL VALOR DEL RESTO R."
140 PRINT :PRINT TAB(3);"LOS RESULTADOS APARECERAN SE-"
150 PRINT "GUN EL ESQUEMA DE LA REGLA DE"
160 PRINT "RUFFINI."
270 PRINT :PRINT :PRINT :PRINT:PRINT
280 FOR I=1 TO 2000:NEXT I
290 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
300 CLS
310 REM ************************
312 REM * INTRODUCCION DEL GRADO *
314 REM * DEL POLINOMIO Y DE SUS *
316 REM *       COEFICIENTES     *
318 REM ************************
320 PRINT :PRINT
330 INPUT "GRADO DEL POLINOMIO P(x); N=";N
340 DIM B(N),C(N)
345 PRINT :PRINT
350 FOR I=N TO 1 STEP -1
360 PRINT "COEFICIENTE DE X^";I;":";:INPUT B(I)
370 NEXT I
380 INPUT "TERMINO INDEPENDIENTE:";B(0)
400 REM ***************************
410 REM * VALOR DEL BINOMIO (x-a) *
412 REM ***************************
420 CLS
430 PRINT :PRINT :PRINT :PRINT :PRINT
440 PRINT :PRINT TAB(6);"VALOR DEL TERMINO (a) DEL"
445 PRINT TAB(6);"BINOMIO (x-a):";
450 INPUT A
460 CLS
500 REM ****************************
505 REM * RESULTADO DE LA DIVISION *
510 REM ****************************
515 PRINT :PRINT
520 PRINT TAB(3);"EL COCIENTE ENTRE EL POLINOMIO"
530 PRINT "P(x) Y EL BINOMIO (x-a) ES:"
540 PRINT:PRINT TAB(10);"[P(x)/(x-a)]"
550 PRINT:PRINT:PRINT
560 LET C(N-1)=B(N)
570 PRINT "COEFICIENTE DE X^";N-1;":";C(N-1)
580 FOR I=N-1 TO 1 STEP -1
590 C(I-1)=C(I)*A+B(I)
600 PRINT "COEFICIENTE DE X^";I-1;":";C(I-1)
610 NEXT I
630 LET R=C(0)*A+B(0)
635 FOR I=1 TO 600:NEXT I
640 PRINT :PRINT "EL RESTO DE LA DIVISION ES R=";R
650 IF R<>0 THEN GOTO 675
655 PRINT :PRINT
658 FOR I=1 TO 1700:NEXT I
660 PRINT :PRINT "LA DIVISION ES EXACTA."
665 PRINT
670 PRINT "X=";A;"ES UNA RAIZ DEL POLINOMIO P(x)."
675 PRINT :PRINT :PRINT
678 FOR I=1 TO 2500:NEXT I
680 INPUT "(PULSA UNA TECLA PARA CONTINUAR)";P$
700 REM **************************
702 REM * EDICION EN PANTALLA DE *
704 REM *   LA REGLA DE RUFFINI  *
706 REM **************************
720 CLS
725 PRINT :PRINT :PRINT :PRINT :PRINT
730 PRINT "      ";
740 FOR I=N TO 0 STEP -1
750 PRINT B(I);"  ";
```

```
760 NEXT I
770 PRINT:PRINT :PRINT:PRINT
780 PRINT A;"  ";
790 FOR I=N TO 0 STEP -1
800 PRINT C(I)*A;"  ";
810 NEXT I
820 PRINT:PRINT
830 PRINT "*********************************"
835 PRINT
840 PRINT "      ";
850 FOR I=N-1 TO 0 STEP -1
860 PRINT C(I);"  ";
870 NEXT I
880 PRINT R;"=R"
890 END
```

Modificaciones para otros equipos

Tienes que hacer las modificaciones siguientes:

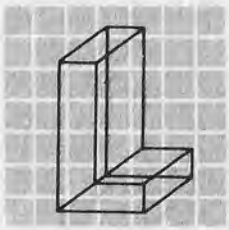
ZX-SPECTRUM

890 GOTO 9999

COMMODORE

50 PRINT CHR$ (147)
300 PRINT CHR$ (147)
420 PRINT CHR$ (147)
460 PRINT CHR$ (147)
720 PRINT CHR$ (147)

Ordenadores típicos de la segunda generación (STRETCH, LARC y GAMMA 60)



OS años cincuenta supusieron un enorme desarrollo de la Informáica. Los ordenadores comenzaron a venderse en serie, accediendo a ellos un entorno de empresas mucho mayor que en años anteriores, en que sólo las muy potentes económicamente podían disponer de ellos. Sin embargo, con las nuevas tecnologías a transistores, la fiabilidad ganó, los consumos bajaron y con ellos los precios y los problemas. Las empresas se atrevieron a fabricar en serie y empezaron a crear modelos de ordenadores, intentando cubrir todas las necesidades de los usuarios. La idea era comenzar a diseñar gamas de productos, para todo tipo de usuarios, desde pequeño nivel hasta complicados sistemas. Esta misma idea es la que aplican las casas fabricantes de automóviles, en las que cada marca fabrica desde el sencillo utilitario sin complicaciones hasta el sofisticado automóvil lleno de prestaciones, y a un precio muy superior. Se trata de captar a todo tipo de usuarios. Con los modelos pequeños se ganan clientes fundamentalmente, con los especiales, prestigio. No hay que olvidar que los pequeños clientes pueden ir creciendo. Las empresas dedicadas a la fabricación y comercialización de ordenadores también comenzaron a seguir esta pauta, creando equipos sencillos que generalmente eran compatibles con otros de superiores prestaciones, siguiendo una gama casi continua. La idea que intentaban vender es la siguiente: un técnico de la casa podría estudiar, junto con el usuario, sus necesidades, para proponerle un equipo a su medida, pudiendo pasar a ampliarlo cuando haya cubierto estas necesidades inmediatas, y considere que puede mecanizar otras tareas de su empresa. Los programas escritos para los distintos equipos de la misma casa, sin embargo, no siempre eran totalmente compatibles y requerían algunas modificaciones. Pero, además, existía una total incompatibilidad entre las distintas marcas, lo que obligaba a seleccionar muy bien la casa fabricante, al comienzo de la mecanización de la compañía, con el riesgo de estar siempre «vendido» a comprar en el mismo sitio.

Otra característica común de la segunda generación es que cada vez se iba tendiendo más a realizar procesos secuenciales en cinta. Este sistema de proceso es bastante útil para la impresión de informes, y para otros tipos de proceso, pero no es eficaz para consultas.

Tampoco hemos hablado mucho de los lenguajes de programación. Estos se fueron desarrollando poco a poco, pero hasta el momento son lenguajes simbólicos de bajo nivel, es decir, muy ligados al funcionamiento de la máquina.

Von Neumann siempre había considerado que el lenguaje más universal para los ordenadores era el lenguaje máquina. Sin embargo, la tendencia que ha ido tomando fuerza con el tiempo es que los ordenadores modernos utilicen lenguajes cada día más sofisticados, en suma, mas próximos al usuario, y no a la máquina.

El gobierno de los Estados Unidos sacó a concurso para la Comisión de Energía Atómica la compra de un ordenador que fuera mucho más rápido de lo que hasta entonces se conocía. Naturalmente, Univac e IBM participaron en el concurso. Con la nueva técnica de los transistores se trataba, entre otras cosas, de obtener velocidades medias en nanosegundos (hasta el momento se medían en microsegundos). IBM diseñó para ello el IBM 7030, llamado también Stretch, en 1960, y Univac en LARC (Livermore Automatic Research Computer), unos meses antes que el IBM. (El Livermore Automatic Research Computer había tomado este nombre debido a la localidad

de Livermore, California, donde se encontraba la entidad receptora del nuevo ordenador. Poco tiempo después, IBM sacó al mercado otro ordenador de mejores características que el 7030, aunque dentro de la misma línea. Veamos con algún detenimiento estas máquinas, que fueron típicas de la época que estamos intentando describir.

Tanto el LARC como el Stretch (e incluso el Gamma 60 de la Bull, en Francia) eran ordenadores especializados para la resolución de problemas científicos. Por tanto, se habían diseñado considerando las distintas opciones, y seleccionando las que más se adaptaran a estas necesidades. En los primeros años de la segunda generación, había una idea bastante clara y general de que las máquinas desarrolladas para funcionar en el mundo empresarial debían ser preferiblemente decimales, y basadas en los caracteres. Para los diseños de ordenadores de tipo científico, las tendencias eran distintas. Entre sus diseñadores existía una fructífera diversidad de opiniones. Unos consideraban que era más práctico hacer que la máquina fuera decimal, mientras que otros muchos consideraban que si utilizaba el sistema binario, se simplificarían mucho los cálculos. Pero los sistemas de numeración adoptados no eran la única diferencia entre unos ordenadores y otros. El tipo de memoria auxiliar también era un elemento a considerar, junto con los dispositivos de entrada-salida. No es lo mismo tener que trabajar con cantidades ingentes de información que con un volumen de datos de tipo medio. También hay que considerar la precisión, que en algunas tareas puede ser necesario que alcance cotas de precisión muy altas. Resumiendo, se puede optar por una variedad de posibilidades, intentando minimizar las desventajas de cada una de ellas.

El LARC de Univac comenzó a trabajar en sistema decimal. Esta decisión se tomó, no sin considerar con todo cuidado la posibilidad del binario. Sin embargo, como el volumen de datos era enorme, y como resultado, las salidas, se pensó que las conversiones binario-decimal serían de tal volumen, que la ganancia de tiempo no se podría apenas aprovechar, debido a las enormes pérdidas de tiempo en las conversiones de un sistema a otro. Además, existía el problema de los redondeos, que siempre producía errores que debían tenerse en cuenta, y que era difícil reducir.

Los diseñadores de IBM, con los mismos argumentos, llegaron a conclusiones distintas.

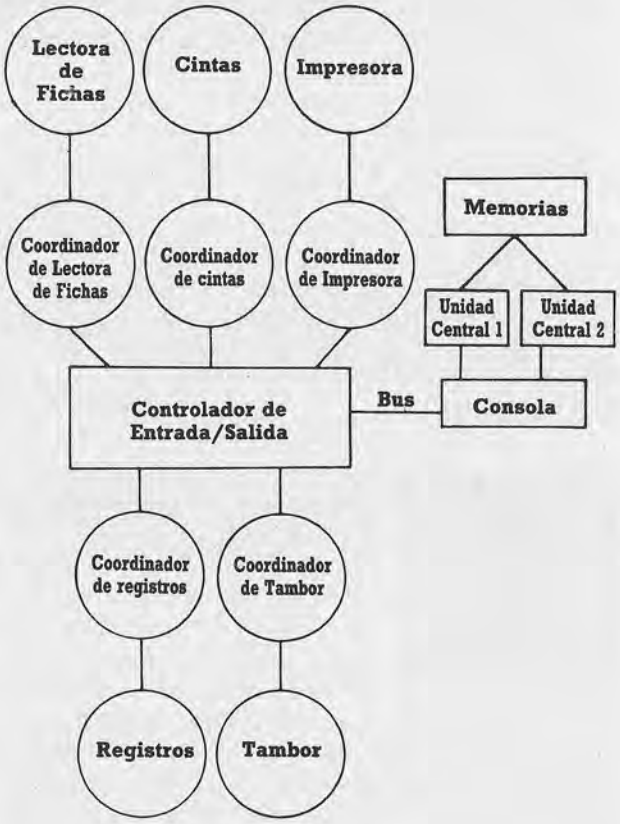


Fig. 1.

Se decidieron por el binario, y esta decisión ha afectado a la casi totalidad de los ordenadores que se han ido fabricando. Es seguro que si la decisión no la hubiera tomado IBM, la habría tomado otra compañía cualquiera de ordenadores, en ese momento o en otro. El hecho es que a partir de esa fecha el sistema binario se ha ido imponiendo hasta desbancar totalmente al decimal. Sus ventajas fundamentales son que simplifica notablemente la estructura de la máquina, el cómputo es mucho más rápido, el control mucho más sencillo y la representación de la información se hace mucho más compacta.

Veamos sucintamente la estructura del Stretch de IBM. Al estar concebida como máquina para uso científico, podía realizar operaciones numéricas y no numéricas. Para ello, los ingenieros que la diseñaron organizaron la palabra básica de 64 bits, formada a su vez por 8 grupos de 8 bits, a los que más tarde se llamaría bytes. En cuanto al código para crear el alfabeto, el que se utilizó era de 8 bits para poder obtener 256 caracteres o posibilidades distintas, ya que los alfabetos con que funcionaban las últimas máquinas eran generalmen-

te de 100 letras y símbolos, siendo relativamente escasos e incómodos. De los 256 se tomaron únicamente 120, ya que tomar más hubiera exigido una impresora muy especial, y no tenía tampoco demasiado interés. Volvemos a insistir en que la máquina utilizaba código en binario. Esta organización del código en elementos de 8 bits, a los que a partir de ese momento se llamará bytes, fue muy útil, tanto como para haberse mantenido hasta nuestros días.

El sistema de almacenamiento del Stretch de IBM era muy avanzado. Es natural, ya que uno de los problemas fundamentales de los ordenadores científicos suele ser el enorme cúmulo de datos, que dificulta todas las tareas. En la actualidad existen nuevas técnicas para abordarlo, utilizando ordenadores paralelos, con estructuras pipeline, etc. Sin embargo, este problema todavía no está totalmente resuelto. Como es primordial disponer de una buena memoria. IBM dotó a su modelo Stretch de una memoria formada por una pila de discos magnéticos, cada uno de los cuales era independiente, disponiendo de su propio brazo de lectura/escritura. También contaba con un sistema llamado TRACTOR de manejo de cintas magnéticas. Se trataba, pues, de sistemas muy complejos y rápidos, que podían leer/escribir a velocidades muy altas (1,5 millones de caracteres por segundo).

En la serie 360 de IBM se utilizaban palabras «simples» o de cuatro bytes, o «dobles» o de ocho bytes. Las cadenas podían ser numéricas, o de caracteres, pero todas ellas formadas por bloques de 8 caracteres (o bytes). Con este sistema de organizar las palabras, los direccionamientos se hacían mucho más sencillos.

El ordenador Gamma 60 de Bull disponía de una organización de las palabras semejante, aunque las cadenas que se utilizaban eran de 24 bits, algo grandes para codificar los caracteres y algo pequeñas para utilizarlas como palabra básica. Con el tiempo se vio que existían sistemas de codificación mejores.

La precisión, en estos ordenadores científicos, es un problema fundamental. La introducción de la coma flotante ayudó mucho a dominar el problema. Para obtener precisión múltiple se podía acceder a un conjunto especial de instrucciones.

Finalmente, el Stretch disponía de sistemas control de interrupciones muy sofisticados que permitía el multiproceso. Para la recuperación de los errores existía, además de la consola normal para el operador, otra para el equipo de mantenimiento. Este podía utilizar en su análisis de la máquina todo un complejo programa de comprobaciones que ayudaba a detectar los problemas. Estos últimos, en algunos casos, podían subsanarse directamente, sin tener, por ejemplo, que detener por ello la ejecución de algún programa. Además, el equipo se entregaba junto con unos

¿Sabía usted que...

En alguna ocasión la utilización de un ordenador puede resultar carísima, incluso para todo un banco?

Efectivamente. El 24 de noviembre de 1985, uno de los bancos más importantes de Nueva York tuvo que pagar más de un millón de dólares como intereses al Banco del Estado (Federal Reserve), que se hubiera evitado si esta entidad no hubiera dispuesto de servicios informatizados a tan alto nivel.

En las fechas de que hablamos, la mayoría de las entidades bancarias de los Estados Unidos pertenecía a un servicio de compensación bancaria electrónica. Al finalizar la actividad en Wall Street, hacia las seis y media de la tarde, comienza la compensación global de los bancos afiliados a dicho servicio. La norma es que aquel que no pueda «satisfacer» su deuda, si al final de la jornada queda deudor, quedará automáticamente excluido del servicio (naturalmente, podemos imaginarnos el enorme perjuicio que ello supone para un gran banco, con miles de operaciones diarias).

Con el antiguo sistema de compensación, que se llevaba a cabo sobre las diez de la mañana, las entidades, si eran deudoras, podían recabar el capital necesario del mercado monetario para hacer frente sin problemas a la situación de cada día. Sin embargo, con el nuevo sistema informatizado, mucho más cómodo y simple, si una entidad necesita dinero a partir de las seis y media de la tarde, el problema es bastante más difícil de resolver.

En esta situación se encontró en aquella fecha uno de los más importantes bancos norteamericanos, que tuvo que recurrir a la Reserva Federal para cubrir su déficit de más de veinticinco millones de dólares. (Bien es cierto que si el sistema informático del propio banco hubiera funcionado como es debido, a pesar de las dificultades hubiera podido disponer del dinero de maniobra necesario, al menos con un coste menor.) No hay duda que una sociedad informatizada requiere la revisión de leyes y normas periódicamente.

programas de diagnóstico, que facilitaban enormemente la identificación de las averías. Estos sistemas de detección de fallos y averías eran de lo más avanzado. Estaban basados en otros diseñados por la IBM para sus ordenadores del proyecto SAGE (Semi Automatic Ground Environment) de defensa antiaérea norteamericana. Como es de imaginar, en el proyecto SABE, la fiabilidad y la rápida reparación de errores eran objetivos fundamentales.

Se vendieron ocho unidades del ordenador Stretch. El precio puede parecer exorbitante (ocho millones de dólares), pero es importante recordar los enormes gastos en investigación de las compañías fabricantes. Los nuevos modelos, menos probados, exigían gastos mucho mayores, tanto en comprobaciones y otras pruebas, etc., como en el desarrollo de sistemas operativos sofisticados, que controlaran adecuadamente todas las innovaciones introducidas a las máquinas. También era necesario desarrollar los compiladores para los nuevos lenguajes de programación que iban apareciendo, etc. No pensamos que las grandes compañías han sido instituciones benéficas interesadas únicamente en el progreso de la Humanidad. Sin embargo, al buscar su beneficio, han ayudado a avanzar a la técnica en su lento y dificultoso caminar.

Control del proceso deductivo

Se suele utilizar el término inglés «backtracking» para designar un proceso de «vuelta atrás» que se realiza con cierta frecuencia en los procesos de búsqueda. En efecto, el esquema de examen de la base de conocimiento no suele llevar un camino en una sola dirección, sino que, en ocasiones, llega a un punto en que debe «volver atrás» y seguir por otro camino o bien detener su búsqueda.

El proceso de «vuelta atrás» es sumamente interesante y añade mucha flexibilidad al esquema de búsqueda, pero requiere un conjunto adicional de controles que dificultan la programación del motor de inferencia (hay que guardar información sobre los valores de los hechos en cada paso para retomar la situación —hechos demostrados y no demostrados, reglas utilizadas y reglas de las que no se ha hecho uso, etc.— en el punto a donde se vuelve para tomar un camino alternativo).

Utilización de lógica de primer orden

Si en lugar de utilizar la lógica de proposiciones se hace más sofisticado el entorno de representación del conocimiento pasando a utilizar lógica de primer orden (cálculo con predicados), la representación de datos es más flexible, pero el control del proceso resulta más complicado. En efecto, la introducción de variables en las reglas de producción supone abordar un verdadero riesgo de «explosión» de hechos demostrados o por demostrar. En los casos más sencillos las proposiciones (hechos) a manejar están dados, y sólo dudamos de su valor («el animal vuela» es verdadero = V, falso = F o desconocido = D); sin embargo, sí decimos «todo ungulado es un mamífero», o, lo que es lo mismo, «si X es ungulado, X es mamífero» (es decir, «ungulado (X) → mamífero (X)») para sacar conclusiones, debemos «instanciar» o particularizar la variable «X» (en principio, cualquier animal): y así preguntarnos si será verdad o no que ungulado (loro), ungulado (jirafa), ungulado (avestruz), ungulado (murciélago), ungulado (cebra)... Además, como normalmente no se producirá la instanciación de la regla para todos los valores posibles, sino sólo para aquel o aquellos que en cada momento nos interesa, las reglas ya no son (como en el caso del cálculo de proposiciones) de utilización una única vez, sino que en cada paso deductivo se pueden utilizar, con lo que hay que controlar no si la regla se ha utilizado o no, sino si ha sido usada con tal o cual variable.

Además, en cada regla puede haber varias variables que se particularizan independientemente unas de otras. Concretamente, si en una regla aparecen cuatro variables (caso usual en un sistema real), y disponemos de cincuenta hechos conocidos en el momento considerado, se pueden probar 50 elevado a 4 (más de 6 millones) de combinaciones diferentes; ya se comprende que, por muy potente que sea el equipo sobre el que se esté trabajando, la búsqueda exhaustiva es impensable en un caso como el considerado. Es necesario establecer un mecanismo automático de conducción del proceso deductivo. En cada momento, el orden (dinámico) en que conviene evaluar las premisas para localizar el conjunto de valores (cuatro en el caso que acabamos de proponer) que hay que seleccionar para aplicar la regla correspondiente depende del contexto y/o de la propia estructura de la regla.

El ciclo de base de un motor de inferencia suele tener tres fases: localización de las reglas utilizables en ese momento, elección de la regla a aplicar entre las posibles (en el caso de la lógica de predicados que estamos considerando es el momento de instanciación de las variables para elegir no sólo la regla a utilizar, sino la n-tupla de valores con los que se va a usar) y realización de la deducción propiamente dicha con la regla y valores considerados.

Las estructuras de control utilizadas en

los diferentes sistemas son muy diferentes (elección en función del interés de la regla según el objetivo a obtener, elección según un juego de hechos considerados básicos, elección de la primera regla encontrada, búsqueda exhaustiva, etc.).

Uno de los sistemas más interesantes es el utilizado en el generador de sistemas expertos SNARK: en la etapa primera de detección de las reglas a utilizar se considera para cada regla un conjunto de «palabras clave» que son las *propiedades* o *valores* significativos que aparecen en la regla y se marca la regla como inutilizable (en ese momento) si alguna de sus palabras-clave no aparece en la base de hechos; si, por el contrario, todas las palabras-clave de la regla aparecen al menos una vez en la base de hechos se anota que esa regla es utilizable en ese ciclo.

A continuación se van examinando las reglas «utilizables», para la localización de la «n-tupla» de valores que permitirán activar esa regla.

Para ello, considera tres tipos diferentes de premisas y selecciona la regla según el tipo de premisas que aparecen en cada una.

Los tres tipos son:

Tipo 1. «propiedad N (X) = a», y la instanciación consiste en localizar un elemento incógnito «X» que cumpla que su «propiedad N» valga «a». Este elemento «X» (candidato provisional a ser uno de los valores seleccionados) se llama «zombi», en el argot.

Tipo 2. «propiedad N (Z) = X», donde «Z» es un elemento (zombi) resultado de una instanciación anterior. X es el nuevo zombi a localizar.

Tipo 3. «propiedad N (X) = Y», hay que localizar simultáneamente una pareja de valores que cumpla la expresión.

Se selecciona la regla según el tipo de premisas que aparecen en ellas (primero tipo 1, luego tipo 2, por último, el tipo 3) y, en caso de igualdad de tipo de premisas, se elige aquella cuya propiedad aparece mayor número de veces en la base de hechos.

Se recorre la base de hechos seleccionando (encadenando) por *valor* o *propiedad* y se toman los objetos que se adecúan al modelo de búsqueda que estamos realizando (se va creando una pila con los valores obtenidos). A continuación se «repercuten» los valores seleccionados sobre las premisas no evaluadas y sobre la parte de conclusión de la regla. De esta implicación pueden surgir instanciaciones nuevas o cambio de tipo de premisas, que son tomados en cuenta en el siguiente ciclo.

Si aparece una contradicción con la base de hechos (que se ha utilizado) o no se ha podido realizar elección alguna de regla, se vuelve atrás, se saca de la pila la anterior selección hecha y se continúa el proceso con ese nuevo valor.

Naturalmente, la estrategia descrita es un caso particular (interesante, eso sí) de un motor de inferencia (cualquier definido sobre un paquete SNARK) y las posibilidades que aparecen en distintos sistemas basados en el conocimiento, son muy variadas. Hemos puesto este ejemplo para mostrar al lector cómo se complica la gestión del proceso deductivo con la introducción de la lógica de primer orden.

Aplicación a otros esquemas de representación

Aunque el esquema de representación no sea la lógica, los sistemas de producción siguen siendo un procedimiento válido de deducción. Podemos pensar en un esquema de representación de tipo atributo-valor para el objeto: «ir al teatro» (*). Se utilizarán las parejas correspondientes a los siguientes atributos: «Distancia» (para representar la distancia en km de mi casa al teatro), «Tiempo» (tiempo de que disponemos hasta el comienzo de la representación, en minutos), «Medios» (procedimiento genérico para ir al teatro; puede ser conducir o andar), «Situación» (se refiere a la ubicación de mi casa; puede ser: afueras o centro), «Meteorología» (que representa el tiempo atmosférico que hace; puede ser: bueno o malo) y «Acción» (modo efectivo por el que iré al teatro; se consideran cuatro posibles acciones de tomar: pedir-un-taxi, ir-en-coche, abrigarse-e-ir-andando e ir-andando).

Partimos de la siguiente base de conocimientos.

Regla 1. Si Distancia es más de 10 km → Medios es conducir.

Regla 2. Si Distancia es más de 2 km, pero menos de 10 km, y Tiempo es menos de 20 min. → Medios es conducir.

Regla 3. Si Distancia es más de 2 km, pero menos de 10 km, y Tiempo es más de 20 min. → Medios es andar.

* Seguimos un ejemplo presentado por Harmon y King en su libro *Expert Systems*. Ed. John Wiley & Sons, 1985.

ATRIBUTO	VALOR
Distancia	Más de 10 km
Distancia	Menos de 10 km y más de 2 km
Tiempo	Mas de 20 min
Tiempo	Menos de 20 min
Situación	Afueras
Situación	Centro
Meteorología	Buena
Meteorología	Mala

ATRIBUTO	VALOR
Medios	Conducir
Medios	Andar

ATRIBUTO	VALOR
Acción	Pedir taxi
Acción	Ir en coche
Acción	Abrigarse e ir andando
Acción	Ir andando

Fig. 1.

Regla 4. Si Medios es conducir y Situación es afueras → Acción es pedir-un-taxi.

Regla 5. Si Medios es conducir y Situación es centro → Acción es ir-en-coche.

Regla 6. Si Medios es andar y Meteorología es mala → Acción es abrigarse-e-ir-andando.

Regla 7. Si Medios es andar y Meteorología es buena → Acción es ir-andando.

Si tratamos de averiguar «cómo ir al teatro», el objetivo es saber qué valor debe tomar el atributo acción. Se puede comenzar con encadenamiento hacia atrás, viendo que el objetivo Acción (con diferentes valores asociados) aparece como consecuencia en las reglas 4, 5, 6 y 7.

Para saber si se puede deducir el valor «pedir-un-taxi» del atributo Acción hay que saber si Medios tiene el valor «conducir» (o no) y si situación tiene el valor «afueras» o no.

Para averiguar el valor de Medios debemos acudir a las reglas 1, 2 y 3, que tienen este atributo como consecuente. De la regla primera obtenemos la necesidad de saber la distancia a que se encuentra el teatro de nuestra casa: el sistema lo preguntará y si la respuesta es 5 km, por ejemplo, es decir, «menos de 10 km, pero más de 2 km» no se puede activar la regla 1. Toma la siguiente posibilidad de obtener un valor para el atributo. Medios (objetivo de búsqueda actual). La regla 2 le «sugiere» preguntar por el tiempo de que disponemos para ir al teatro. Si contestamos que tenemos 10 min de tiempo, se dispara la regla 2 y pasamos a saber que Medios vale «conducir».

ATRIBUTO	VALOR
Distancia	Más de 10 km
Distancia	Menos de 10 km y más de 2 km
Tiempo	Más de 20 min
Tiempo	Menos de 20 min
Situación	Afueras
Situación	Centro
Meteorología	Buena
Meteorología	Mala

ATRIBUTO	VALOR
Medios	Conducir
Medios	Andar

ATRIBUTO	VALOR
Acción	Pedir taxi
Acción	Ir en coche
Acción	Abrigarse e ir andando
Acción	Ir andando

Fig. 2.

Dependiento de cómo esté previsto el proceso de control de deducciones, ahora se aplicará el dato obtenido, en encadenamiento hacia adelante, para llegar a concluir valores de Acción o no; y, por otro lado, seguirá examinando las restantes posibilidades del atributo Medios (según la regla 3) o no; puede que se admitan varios valores de un atributo (en este caso significarían opciones alternativas para ir al teatro) o que se consideren contradicciones y se rechacen.

Una vez sabido que Medios es «conducir», el sistema preguntará la situación de la casa (regla 4). Si decimos que «afueras», deducirá el valor «pedir-un-taxi» para Acción. Si sólo queríamos un valor (o el sistema está preparado para dar un solo valor de cada atributo) el proceso deductivo concluye. En caso contrario, el proceso continúa observando otros posibles valores del atributo Acción, pero las reglas restantes (5, 6 y 7) no pueden activarse (por la situación la 5 y por los medios la 6 y la 7). El proceso concluye.

Como se ve, el modo de trabajo es el mismo ya descrito anteriormente, pero aquí se maneja globalmente el concepto de atributo (y se examinan sus posibles valores). Esto suele suponer el análisis global (aunque sucesivo, naturalmente) de varias reglas y suele dirigir el *backtracking* de tal modo que la búsqueda se circunscribe a los valores de un atributo antes de pasar a otro atributo o a otra regla genérica.

TERMINOLOGIA

Adquisición de conocimiento. Proceso de extracción, estructuración y organización del conocimiento, partiendo de alguna fuente de conocimiento (generalmente una persona experta), y que puede utilizarse en un programa.

Arbol. Gráfico en el que sólo existe un camino entre dos nodos distintos.

Base de conocimientos. Una parte del sistema basado en el conocimiento, o sistema experto que contiene el conocimiento sobre el dominio de que se trate.

Base de datos. Conjunto de hechos, aseveraciones y conclusiones que se utilizan para concordar las partes condicionales (IF) de las reglas, en un sistema basado en reglas.

Ciclo de reconocimiento. Bucle interactivo que tiene tres fases: concordancia, resolución de conflictos y disparo de las reglas.

Distintivo. Instancia de un tipo. También puede ser un átomo único que puede utilizarse como etiqueta, o bien un símbolo utilizado en el algoritmo de concordancia de Rete para representar un elemento de la memoria de trabajo.

EMYCIN. La primera herramienta de construcción de sistemas expertos. EMYCIN deriva del sistema experto MYCIN. Después de desarrollar MYCIN, los diseñadores consideraron que podían extraer el conocimiento médico específico de MYCIN (es decir, el MYCIN esencial). El módulo-concha (shell) resultante consiste en una máquina de inferencias de encadenamiento hacia atrás, un director de consultas, y varias ayudas de adquisición de conocimiento. Esta «concha», o herramienta, podía combinarse con otra base de conocimientos para crear un nuevo sistema experto.

Espacio de un problema. Gráfico en el que los nodos representan todos los estados posibles de las soluciones parciales o completas del problema, y los arcos representan los operadores que transforman un estado en otro. La búsqueda de la solución al problema que se está considerando se representa estableciendo un camino a partir del nodo que representa el estado inicial, hasta el nodo que representa el estado objetivo.

Ingeniero del conocimiento. Aquella persona que diseña y construye el sistema experto. Suele ser un experto en la aplicación de métodos de inteligencia artificial.

Jerarquía de herencia. Estructura dentro de una red o sistema de estructura semánticos, que permite a los elementos inferiores en la red heredar propiedades de los elementos más altos en dicha red.

Lenguaje de contexto libre. Lenguaje formal en el que cada sentencia se puede generar mediante una gramática en la que el lado izquierdo de cada regla de reescritura consiste en un único símbolo no terminal.

Ligaduras parciales. Conjunto de elementos de la memoria de trabajo y ligaduras que constituyen una ligadura consistente para una secuencia prefijada de elementos de condición de una regla.

Metaconocimientos. En un sistema experto, conocimiento de cómo el sistema opera o razona, por ejemplo, el conocimiento de cómo éste utiliza y controla el conocimiento del dominio. Conocimiento sobre el conocimiento.

Métodos orientados al objeto. Métodos de programación basados en la utilización de elementos llamados objetos, que se comunican unos con otros a través de mensajes.

Programa de un sistema de producción. Programa de aplicación escrito en un lenguaje de sistema de producción y en el que la mayor parte del problema se lleva a cabo disparando las reglas.

Razonamiento dirigido mediante supuestos. Estrategia de resolución de problemas que genera hipótesis sobre los acontecimientos que se espera ocurran, centrándose en procesar las tareas relacionadas

con esos acontecimientos. La actividad de esta estrategia contrasta con la pasividad del razonamiento dirigido a los datos.

Red semántica. Estructura de datos para representar el conocimiento declarativo. Esta estructura es un gráfico en el que los nodos representan conceptos (elementos diversos), y los arcos (que pueden llevar etiquetas) representan las relaciones entre los conceptos.

Sistema «runtime» o versión «runtime». Las herramientas de construcción de los sistemas basados en el conocimiento permiten al usuario crear y ejecutar varias bases de conocimiento. Con una única herramienta, el usuario puede crear docenas de bases de conocimiento. Dependiendo del problema al que se encare el usuario, cargarán una base de conocimientos apropiada y llevarán a cabo consultas. Con esta herramienta, el usuario puede modificar fácilmente una base de conocimientos. Las compañías que desarrollan bases de conocimiento específicas no desean que el usuario «cargue» la base de conocimientos, ni que sea capaz de modificarla. Cuando una herramienta de construcción de un sistema experto se modifica para incorporar una base de conocimientos específica, y para desactivar algunas de las características de programación, el sistema resultante se dice que es un «runtime system» o una versión «runtime». (A veces se llaman sistemas de «solo ejecución»).